

09/926320

PCT/JP 00/02512

日 本 国 特 許 庁

PATENT OFFICE
JAPANESE GOVERNMENT

17.04.00

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application:

1999年 4月22日

REC'D 05 JUN 2000

出 願 番 号
Application Number:

平成11年特許願第115047号

WIPO

PCT

出 願 人
Applicant (s):

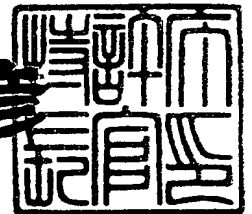
関 一

PRIORITY
DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

2000年 5月19日

特 許 庁 長 官
Commissioner,
Patent Office

近 藤 隆 彦



出証番号 出証特2000-3037035

【書類名】 特許願

【整理番号】 R002

【あて先】 特許庁長官殿

【国際特許分類】 G06F 15/00

【発明者】

 【住所又は居所】 愛媛県松山市道後喜多町 4 番 3 8 号

 【氏名】 関 一

【特許出願人】

 【識別番号】 598003070

 【住所又は居所】 愛媛県松山市道後喜多町 4 番 3 8 号

 【氏名又は名称】 関 一

【手数料の表示】

 【予納台帳番号】 057509

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

 【物件名】 図面 1

 【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書
【発明の名称】 計算機システム
【特許請求の範囲】

【請求項 1】

各々のエントリにデータが書き込まれるようになっている統合レジスタ・ファイル（6）と、

各々のエントリに該統合レジスタ・ファイル（6）のエントリのアドレスが書き込まれるようになっている前進ポインタ・スタック（3）と、

各々のエントリに個々の命令の内容が書き込まれるようになっている F I F O（First In First Out）キューの構成となっている命令バッファ（5）と、

各々適当な数のリザーベーション・ステーションを備える機能ユニット群と、

該統合レジスタ・ファイル（6）及び該機能ユニット群の間でデータを該統合レジスタ・ファイル（6）のエントリのアドレスと共に分配するようになっている共通データ・バス（8）とを具備し、

デコードされた命令においてオペランド・スタックに対するポップ操作が規定されている場合には、ポップすべき語数と同じ数だけ該統合レジスタ・ファイル（6）のエントリのアドレスを該前進ポインタ・スタック（3）からポップし、

デコードされた命令においてオペランド・スタックに対するプッシュ操作が規定されている場合には、プッシュすべき語数と同じ数だけ割り付けられていない該統合レジスタ・ファイル（6）のエントリを割り付け、該割り付けた該統合レジスタ・ファイル（6）のエントリのアドレスを該前進ポインタ・スタック（3）にプッシュし、

デコードされた命令の内容を、ポップ／プッシュ操作を伴う命令の場合にはポップ／プッシュされる該統合レジスタ・ファイル（6）のエントリのアドレスと共に、該命令バッファ（5）に書き込み、

命令の種類に応じて、必要な場合には、該命令バッファ（5）に書き込まれる命令の内容を、該命令を実行する機能ユニットの命令の内容を保持していないリザーベーション・ステーションにも書き込み、

該前進ポインタ・スタック（3）からエントリ・アドレスがポップされる該統合

レジスタ・ファイル（６）のエントリの各々の内容が読み出され、データが既に書き込まれている場合には、エントリ・アドレスとデータが該共通データ・バス（８）に載せられるようになっており、

該リザベーション・ステーションの各々において、命令の内容を保持している場合、そこに書き込まれているソース・データを保持すべき該統合レジスタ・ファイル（６）のエントリのアドレスと該共通データ・バス（８）を通じて送られてくる該統合レジスタ・ファイル（６）のエントリのアドレスが比較され一致すればデータが取り込まれ、必要なソース・データが揃った後に該命令の実行が開始され、

該機能ユニット群の各々において、デコードの際該前進ポインタ・スタック（３）に該統合レジスタ・ファイル（６）のエントリのアドレスがプッシュされるような命令の実行の結果得られたデータを該プッシュされた該統合レジスタ・ファイル（６）の該エントリの該アドレスと共に該共通データ・バス（８）に載せ

、
該統合レジスタ・ファイル（６）においては、該共通データ・バス（８）を通じて送られてくる内容に基づきデータの書き込みを行うことによって、スタックマシンの機械語で記述されたプログラムを実行する計算機システム。

【請求項２】

各々のエントリに該統合レジスタ・ファイル（６）のエントリのアドレスが書き込まれるようになっている完了ポインタ・スタック（４）を具備し、

該命令バッファ（５）におけるキューの先頭のエントリに保持されている命令の完了が可能である、あるいはそうなり、該キューの該先頭のエントリの内容に基づき、該保持されている命令がデコードされた際の該前進ポインタ・スタック（３）の動作を再現すべく該完了ポインタ・スタック（４）を操作し、該キューから該先頭のエントリを除外し、

ポップ操作によって該完了ポインタ・スタック（４）におけるアドレスの保持が無くなった該統合レジスタ・ファイル（６）のエントリの割り付けを解除するようになっている請求項１記載の計算機システム。

【請求項３】

データ・キャッシュ（11）とデータ・バッファ（12）を具備し、演算を実行するようになっている演算ユニット（80、81）と該データ・キャッシュ（11）及び該データ・バッファ（12）にアクセスできるようになっているロード／ストア・ユニット（83）とを該機能ユニット群に含む請求項2記載の計算機システム。

【請求項4】

割り付けられていない該統合レジスタ・ファイル（6）のエントリのアドレスを保持するフリー・リストを具備し、

初期状態においては、該統合レジスタ・ファイル（6）の全てのエントリのアドレスが該フリー・リストに登録されており、

該統合レジスタ・ファイル（6）のエントリを割り付ける必要がある場合に、該フリー・リストから割り付けられていない該統合レジスタ・ファイル（6）のエントリのアドレスを取り出し、

割り付けが解除された該統合レジスタ・ファイル（6）のエントリのアドレスを該フリー・リストに登録するようになっている請求項2記載の計算機システム。

【請求項5】

各々のエントリに該前進ポインタ・スタック（3）の内容が書き込まれるようになっている前進ポインタ・スタック履歴ファイル（3a）を具備し、

該統合レジスタ・ファイル（6）が各々のエントリに分岐タグも書き込まれるような構成となっており、

命令のデコードの際に、割り付けられる該統合レジスタ・ファイル（6）のエントリに分岐タグを書き込むようになっており、

条件分岐命令がデコードされるごとに、該前進ポインタ・スタック履歴ファイル（3a）の1エントリに該前進ポインタ・スタック（3）の内容を書き込み、分岐タグを変更して、分岐予測に基づく投機的実行を行い、

分岐予測が外れた場合には、その条件分岐命令以降にデコードされた命令を無効にし、該条件分岐命令以降にデコードされた命令に付した分岐タグが書き込まれている該統合レジスタ・ファイル（6）のエントリの割り付けを解除し、該条

件分岐命令がデコードされた際に書き込みの行われた該前進ポインタ・スタック履歴ファイル（3 a）のエントリの内容を、該前進ポインタ・スタック（3）にコピーして、正しい位置の命令から処理を再開することによって、

分岐予測に基づく投機的実行を行うようになっている請求項 2 記載の計算機システム。

【請求項 6】

該前進ポインタ・スタック（3）と該完了ポインタ・スタック（4）が循環型のバッファの構成となっており、

該前進ポインタ・スタック（3）及び該完了ポインタ・スタック（4）で、該統合レジスタ・ファイル（6）のエントリのアドレスを保持する最下位のエントリの内容が一致する場合には、該前進ポインタ・スタック（3）及び該完了ポインタ・スタック（4）において該最下位のエントリにおける該統合レジスタ・ファイル（6）のエントリのアドレスの保持を解除し、該一致する内容で示される該統合レジスタ・ファイル（6）のエントリに書き込まれているデータを該データ・バッファ（1 2）にストア（Spill）することができるようになっており、

該データ・バッファ（1 2）に最後にストア（Spill）したデータに対し、割り付けられていない該統合レジスタ・ファイル（6）の 1 エントリを割り付け、該データを書き込み、該前進ポインタ・スタック（3）及び該完了ポインタ・スタック（4）において、該統合レジスタ・ファイル（6）のエントリのアドレスを保持する最下位のエントリの 1 つ下のエントリに該データが書き込まれる該統合レジスタ・ファイル（6）のエントリのアドレスを保持させることによって、該最後にストア（Spill）したデータの該統合レジスタ・ファイル（6）へのロード（Fill）を行えるようになっている請求項 3 記載の計算機システム。

【請求項 7】

該フリー・リストが F I F O キューの構成となっており、

同時に複数の命令をデコードし、該前進ポインタ・スタック（3）の操作、該統合レジスタ・ファイル（6）のエントリの割り付け及び該命令バッファ（5）の連続する複数のエントリへの命令の内容の書き込みを行う機能と、

同時に該命令バッファ（5）の連続する複数のエントリに書き込まれている内

容に基づき、該完了ポインタ・スタック（４）の操作及び該統合レジスタ・ファイル（６）のエントリの割り付けの解除を行う機能を有する請求項４記載の計算機システム。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】

本発明は、スタックマシンの機械語で記述されたプログラムを高速で処理する新規な構成の計算機システムに関するものである。

【０００２】

【従来の技術】

従来、スタックマシンにおいては、命令の実行は、基本的にプログラム上の順序通り（in-order）に行われるものであった。すなわち、スタックマシンにおける演算命令は、オペランド・スタックからソース・データをポップし、演算を実行し、その演算結果をオペランド・スタックにプッシュするというような動作を指示するものであるが、このような命令の連鎖として書かれたプログラムを逐次的に実行するのである。

【０００３】

【発明が解決しようとする課題】

上述したような従来のスタックマシンにおいては、命令をプログラム上の順序通り（in-order）に実行するので、制御構造が単純なもので済むという利点があるが、処理速度が制約を受けるといった問題点があった。

【０００４】

そこで、スタックマシンの機械語で記述されたプログラムをout-of-orderで処理するような計算機方式が考案された。例えば、日本特公平２－２６００８２号、米国特許第５５２２０５１号や、米国特許第５３３３３２０号及び米国特許第５７６５０１４号におけるプロセッサ要素がある。これらの明細書に示されるプロセッサは、処理性能の向上という点で十分ではない上に、正確な例外処理を保証する上で問題があった。

【０００５】

本発明は、上記問題点を解決するため創案されたものであり、正確な例外処理を保証しつつ、スタックマシンの機械語で記述されたプログラムをout-of-orderでより効率的に処理する計算機システムを提供することを目的としている。

【 0 0 0 6 】

【課題を解決するための手段】

本発明による計算機システムは、データ・キャッシュと、データ・バッファと、各々のエントリにデータが書き込まれるようになっている統合レジスタ・ファイルと、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが書き込まれるようになっている前進ポインタ・スタック及び完了ポインタ・スタックと、各々のエントリに個々の命令の内容が書き込まれるようになっているF I F Oキューの構成となっている命令バッファと、演算を実行するようになっている演算ユニットとデータ・バッファ及びデータ・キャッシュにアクセスできるようになっているロード／ストア・ユニットを含む機能ユニット群と、統合レジスタ・ファイル及び機能ユニット群の間でデータを統合レジスタ・ファイルのエントリのアドレスと共に分配するようになっている共通データ・バスとを具備する。上記機能ユニットの各々は、適当な数のリザーベーション・ステーションを備える。

【 0 0 0 7 】

従来のスタックマシンにおいて、スタックが , word1, word2, word3, word4 (右端がスタックトップ) となっている状態は、本発明による計算機システムにおいて、ポインタ・スタックが , < a > , < b > , < c > , < d > (右端がスタックトップ) で、エントリ・アドレスが < a > , < b > , < c > 及び < d > である統合レジスタ・ファイルの各エントリに、それぞれ word1, word2, word3 及び word4 が保持されている状態に対応する。

【 0 0 0 8 】

本発明の計算機システムにおいては、命令がデコードされるごとに、命令の内容に応じて前進ポインタ・スタック及び統合レジスタ・ファイルを操作すると共に、命令の内容を命令バッファ及び、必要な場合には、適切な機能ユニットの空いているリザーベーション・ステーションに書き込むようになっている。この際、命令に規定されているオペランド・スタックに対するスタック操作が、前進ポイン

タ・スタックに対して同様に適用される。ここで、1語のデータのオペランド・スタックへのプッシュ操作を、本発明の計算機システムにおいてエミュレートするには、そのデータを保持すべき統合レジスタ・ファイルの空いている1エントリを割り付け、そのエントリのアドレスを前進ポインタ・スタックにプッシュすればよい。

【0009】

即ち、デコードされた命令においてオペランド・スタックに対するポップ操作が規定されている場合には、ポップすべき語数と同じ数だけ統合レジスタ・ファイルのエントリのアドレスを前進ポインタ・スタックからポップする。デコードされた命令においてオペランド・スタックに対するプッシュ操作が規定されている場合には、プッシュすべき語数と同じ数だけ統合レジスタ・ファイルの空いているエントリを割り付け、上記割り付けた統合レジスタ・ファイルのエントリのアドレスを前進ポインタ・スタックにプッシュする。さらに、デコードされた命令の内容を、ポップ／プッシュ操作を伴う命令の場合にはポップ／プッシュされる統合レジスタ・ファイルのエントリのアドレスと共に、命令バッファに書き込むようになっている。機能ユニットによる実行の必要な命令の場合には、命令バッファに書き込まれる命令の内容を、適切な機能ユニットの空いているリザーベーション・ステーションにも書き込むようになっている。

【0010】

前進ポインタ・スタックからエントリ・アドレスがポップされる統合レジスタ・ファイルのエントリの各々の内容が読み出され、データが既に書き込まれている場合には、後で、エントリ・アドレスとデータが共通データ・バスに載せられるようになっている。

【0011】

リザーベーション・ステーションに書き込まれた命令に関して、原則として次のような動作が順次行われる。各々のリザーベーション・ステーションで、そこに書き込まれているソース・データを保持すべき統合レジスタ・ファイルのエントリのアドレスと共通データ・バスを通じて送られてくるものが比較され、一致すればデータが取り込まれる。必要なソース・データが揃った後に、その命令の実行が

開始される。デコードの際前進ポインタ・スタックに統合レジスタ・ファイルのエントリのアドレスがプッシュされるような命令の場合、機能ユニットでの実行の結果得られたデータを該プッシュされた統合レジスタ・ファイルのエントリのアドレスと共に共通データ・バスに載せる。統合レジスタ・ファイルにおいては、共通データ・バスを通じて送られてくる内容に基づきデータの書き込みを行う。

【0012】

命令バッファにおけるキューの先頭のエントリに保持されている命令の完了が可能である、あるいはそうになると、そのキューの先頭のエントリの内容に基づき、保持されている命令がデコードされた際の前進ポインタ・スタックの動作を再現すべく完了ポインタ・スタックを操作し、キューからその先頭のエントリを除外し、ポップ操作によって完了ポインタ・スタックにおけるアドレスの保持がなくなった統合レジスタ・ファイルのエントリの割り付けを解除するようになっている。

【0013】

【実施例】

以下に、本発明による計算機システムの一実施例について、図面を参照しながら説明する。なお、以下に述べる本発明による計算機システムの実施例は、Java Virtual Machine (Java VM) で規定されるスタックマシンの基本的な命令をハードウェアで実行するものである。すなわち、データ語長を32ビットとして、これを単位にロード／ストア及び算術論理演算等の演算を行う。従って、例えば、倍長語の間での算術演算は、2語ずつ合せて4語のソース・データをもとに2語の演算結果を生ずる。

【0014】

従来のスタックマシンにおける、語の単位でデータがプッシュ／ポップされるようになっているスタックは、後述するポインタ・スタックと区別するために、以降では、ワード・スタックと呼ぶことにする。

【0015】

Java VM においては、ワード・スタックには、メソッドが呼び出されるごとにフレームが積まれる。各フレームで、下部はローカル変数やパラメータの格納域

、上部はオペランド・スタックとなっている。

【 0 0 1 6 】

Java VM にはもともとハードウェアで実行することを想定していない複雑な命令が含まれるが、以下に述べる本発明による計算機システムの実施例は、次のような基本的な命令をハードウェアで実行するものとする。

(a) 即値データのオペランド・スタックへのプッシュ命令

bipush, sipush, aconst_null, iconst_m1, iconst_<i>, fconst_<f>, lconst_<l>, dconst_<d>

(b) 変数データのオペランド・スタックへのロード命令

ldc1, ldc2, iload, iload_<n>, fload, fload_<n>, aload, aload_<n>, ldc2w, lload, lload_<n>, dload, dload_<n>, iaload, laload, faload, daload, aaload, baload, caload, saload

(c) オペランド・スタック上のデータの変数へのストア命令

istore, istore_<n>, fstore, fstore_<n>, astore, astore_<n>, lstore, lstore_<n>, dstore, dstore_<n>, iastore, lastore, fastore, dastore, aastore, bastore, castore, sastore

(d) 演算命令

(d-1) 算術演算命令

iadd, ladd, fadd, dadd, isub, lsub, fsub, dsub, imul, lmul, fmul, dmul, idiv, ldiv, fdiv, ddiv, irem, lrem, frem, drem, ineg, lneg, fneg, dneg

(d-2) 論理演算命令

ishl, ishr, iushr, lshl, lshr, lushr, iand, land, ior, lor, ixor, lxor

(d-3) 変換演算命令

i2l, i2f, i2d, l2i, l2f, l2d, f2i, f2l, f2d, d2i, d2l, d2f, int2byte, int2char, int2short

(d-4) 比較演算命令

lcmp, fcmpl, fcmpg, dcmpl, dcmpg

(e) オペランド・スタックの操作命令

pop, pop2, dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2, swap

(f) 分岐命令

ifeq, ifnull, iflt, ifle, ifne, ifnonnull, ifgt, ifge, if_icmpeq, if_icm
pne, if_icmplt, if_icmpgt, if_icmple, if_icmpge, goto, goto_w

【0017】

以降、特にことわらない限り、「命令」とは上に挙げた命令のいずれかを意味するものとする。

【0018】

図1は計算機システムのブロック図であって、10は命令キャッシュ、11はデータ・キャッシュ、12はデータ・バッファ、20は命令フェッチ・ユニット、21は命令デコード・発行ユニット、3は前進ポインタ・スタック、3aは前進ポインタ・スタック履歴ファイル、4は完了ポインタ・スタック、5は命令バッファ、6は統合レジスタ・ファイル、7はフリー・リスト、8は共通データ・バス、80及び81は各々演算ユニット0及び1、82は分岐ユニット、83はロード／ストア・ユニット、801、802、811、812、821、822、831及び832はリザベーション・ステーションを表している。

【0019】

次に、本発明実施例の計算機システムの各構成要素ごとにその詳細な構成を説明する。

【0020】

(A) 命令フェッチ・ユニット

命令フェッチ・ユニットは、図示してないプログラムカウンタ（pcレジスタ）を具備しており、命令キャッシュから命令をフェッチし、命令デコード・発行ユニットに渡すようになっている。分岐の予測や分岐の実行も担う。

【0021】

(B) 命令デコード・発行ユニット

命令デコード・発行ユニットは、命令フェッチ・ユニットから渡された命令のデコードを行い、プログラムに含まれる命令がout-of-orderで実行されるように、後述する前進ポインタ・スタック、命令バッファ及び統合レジスタ・ファイル等を設定するための各種信号を発生するようになっている。

【0022】

(C) ポインタ・スタック

ポインタ・スタックは、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが書き込まれるようになっているレジスタ・ファイルで構成されている。

【0023】

従来のスタックマシンにおいて、ワード・スタックが, word1, word2, word3, word4 (右端がスタックトップ) となっている状態は、本発明による計算機システムにおいて、ポインタ・スタックが, <a>, , <c>, <d> (右端がスタックトップ) で、エントリ・アドレスが <a>, , <c> 及び <d> である統合レジスタ・ファイルの各エントリに、それぞれ word1, word2, word3 及び word4 が保持されている状態に対応する。

【0024】

本発明の計算機システムは、前進ポインタ・スタック (A P S ; Advanced Pointer Stack) と完了ポインタ・スタック (C P S ; Completed Pointer Stack) の2つのポインタ・スタックを具備する。

【0025】

本発明の計算機システムにおいては、命令がデコードされるごとに、命令の内容に応じて前進ポインタ・スタック (以下ではA P Sと記す) 及び統合レジスタ・ファイルを操作すると共に、命令の内容を命令バッファ及び、必要な場合には、適切な機能ユニットの空いているリザーベーション・ステーションに書き込むことにより、プログラムに含まれる命令がout-of-orderで実行されるべく設定される。すなわち、前進ポインタ・スタックはデコード・発行済みの全ての命令によるスタック操作を反映している。

【0026】

他方、完了ポインタ・スタック (以下ではC P Sと記す) は、プログラム上の順番で完了済みの全ての命令によるスタック操作を反映するものである。本発明の計算機システムはデータ駆動の原理に基づく out-of-order 実行を可能とするものであるが、完了ポインタ・スタックは、正確な例外処理を保証するため、in-

orderで完了済の全ての命令に基づく状態を構成するために存在するものである。

【 0 0 2 7 】

本発明実施例の計算機システムにおいては、ポインタ・スタック及び統合レジスタ・ファイルによって、ワード・スタックの一番上に積まれたフレームの上部のオペランド・スタックの部分のみが保持されるようになっている。ワード・スタックの残りの部分はデータ・バッファ及びデータ・キャッシュに格納されるようになっている。また、オペランド・スタックが成長して、ポインタ・スタック及び統合レジスタ・ファイルで保持しきれなくなると、後述するように、オペランド・スタックの下部がデータ・バッファにSpillされるようになっている。

【 0 0 2 8 】

各ポインタ・スタックは循環型のバッファの構成となっており、プッシュ・ポインタとボトム・ポインタと呼ぶ2つのレジスタが各々存在する。プッシュ・ポインタは、統合レジスタ・ファイルのエントリのアドレスを保持する最上位のエントリの1つ上を示す。ボトム・ポインタは、統合レジスタ・ファイルのエントリのアドレスを保持する最下位のエントリを示す。ボトム・ポインタの値からプッシュ・ポインタの値を引くことで、ポインタ・スタックに何エントリの空きがあるかがわかる。初期状態においては、プッシュ・ポインタ及びボトム・ポインタの各々の値は共に0となっている。

【 0 0 2 9 】

図2は、本実施例の計算機システムにおける、各ポインタ・スタックと各プッシュ・ポインタ及びボトム・ポインタの関係を示す説明図である。2つのポインタ・スタックAPS3及びCPS4は同数のエントリを有し、各ポインタ・スタックで各々のエントリに下から順に0、1、2、・・・とアドレスが付けられているものとする。縦線が施されているエントリは統合レジスタ・ファイルのエントリのアドレスを保持しているものとする。図2に示すように、プッシュ・ポインタは、APS及びCPSの各々に対して設けられており、それぞれPP_OF_APS及びPP_OF_CPSと名付けている。他方、ボトム・ポインタは1つだけ設けられており、これがAPS及びCPSで共用される。これをBP_OF_PSと名付けている。

【0030】

A P S と C P S の間には、エントリの数だけ比較回路が設けられており、A P S 及び C P S の同じエントリ・アドレスにある（図2において水平に並ぶ）エントリの間でその内容が比較されるようになっている。

【0031】

命令のデコード・発行の際、命令に規定されているオペランド・スタックに対する1語分のプッシュ操作に対応して、割り付けられる統合レジスタ・ファイルの1エントリのアドレスをA P S のPP_OF_APSで示されるエントリに書き込み、PP_OF_APSの値に1を加えるようになっている。逆に、命令に規定されているオペランド・スタックに対する1語分のポップ操作に対応して、PP_OF_APSの値から1を引くようになっている。命令の完了の際のC P S とPP_OF_CPSの動作に関しても同様である。

【0032】

BP_OF_PSで示されるエントリの内容がA P S とC P S で一致する場合には、その一致する内容で示される統合レジスタ・ファイルのエントリに書き込まれている1語分のデータをデータ・バッファにSpillすることができる。その際、BP_OF_PSの値に1を加えるようになっている。逆に、データ・バッファから統合レジスタ・ファイルに1語分のデータをFillするには、データ・バッファからFillすべき1語分のデータを取り出し、それに空いている統合レジスタ・ファイルの1エントリを割り付けてそのデータを書き込み、その統合レジスタ・ファイルのエントリのアドレスをA P S 及びC P S のBP_OF_PSで示されるエントリの1つ下に各々書き込み、BP_OF_PSの値から1を引くようになっている。

【0033】

本実施例の計算機システムは、分岐予測に基づく投機的実行を可能にするために、前進ポインタ・スタック履歴ファイル（以下では「A P S 履歴ファイル」と記す）を具備する。A P S 履歴ファイルの各々のエントリには、A P S の全エントリ及びPP_OF_APSの内容が書き込めるようになっている。

【0034】

(D) 統合レジスタ・ファイル (C R F ; Consolidated Resister File)

統合レジスタ・ファイル（以下ではCRFと記す）は、従来のスタックマシンにおけるオペランド・スタックの内容を、順序不同で保持するものである。

【0035】

図3は、本実施例の計算機システムにおける、CRF6の各々のエントリ6(i)の詳細な構成を示す説明図である。ここで、iはエントリのアドレスである。CRF6の各々のエントリ6(i)はデータ・フィールド61(i)、書込み完了フラグ(WCF, Write Completion Flag)フィールド62(i)、カラー(C, Colour)フィールド63(i)及びビジービット(BB)フィールド64(i)から成っている。

【0036】

実際のCRFのハードウェア上の構成は、上述の各フィールド別に設けられたレジスタ・ファイルの集合体である。

【0037】

CRFの各々のエントリのデータ・フィールドは、1語分のデータが書き込まれる構成となっている。

【0038】

CRFの各々のエントリにおいて、WCFフィールドは、データ・フィールドにデータの書き込みが完了していれば1、完了していなければ0が書き込まれているようになっている。

【0039】

CRFの各々のエントリにおいて、Cフィールドは、そのCRFのエントリが、命令に含まれるプッシュ操作に対応して割り付けられたものであるのか、データ・バッファからのFillの際に割り付けられたものであるのかの区別、前者の場合にはさらに分岐タグが書き込まれるようになっている。本実施例においては、後述するように、分岐タグはAPS履歴ファイルのエントリのアドレスと一定の関係にある。

【0040】

CRFの各々のエントリにおいて、BBフィールドは、そのCRFのエントリがデータを保持すべく割り付けられている状態であれば1、割り付けられていな

い（空いている）状態であれば 0 が書き込まれているようになっている。

【 0 0 4 1 】

（E）フリー・リスト（F L）

フリー・リスト（以下では F L と記す）は、フリーな、即ち、空いている／割り付けられていない（B B フィールドが 0 である）C R F のエントリのアドレスを保持するもので、本実施例においては、循環型の F I F O キューの構成となっている。

【 0 0 4 2 】

初期状態においては、C R F の全てのエントリのアドレスが F L に登録されている。空いている C R F のエントリを割り付ける必要がある場合に、F L からフリーな C R F のエントリのアドレスが取り出される。逆に、C R F のあるエントリの割り付けが解除されれば、そのエントリのアドレスが F L に登録されるようになっている。

【 0 0 4 3 】

（F）命令バッファ（I B ; Instruction Buffer）

命令バッファ（以下では I B と記す）は、未完了の発行済命令を保持するバッファであり、循環型の F I F O キューの構成となっている。

【 0 0 4 4 】

図 4 は、I B の構成を示す説明図である。図 4 において、I B 5 の各々のエントリは下から順に 0、1、2、…とアドレスが付けられているものとし、縦線が施されている I B 5 のエントリは、未完了の発行済命令を保持しているものとする。I B は、ヘッダ・ポインタとトレイル・ポインタと名付けた 2 つのレジスタを具備する。ヘッダ・ポインタはキューの先頭のエントリを、トレイル・ポインタはキューの末尾のエントリの 1 つ後を示す。1 サイクル当たり 1 命令までしか発行／完了を行わないものとするれば、トレイル・ポインタは次に発行される命令の内容を書き込むべきエントリを示し、ヘッダ・ポインタは次に完了されるべき命令の内容が書き込まれているエントリを示す。ヘッダ・ポインタの値からトレイル・ポインタの値を引くことで、I B に何エントリの空きがあるかがわかる。初期状態においては、ヘッダ・ポインタ及びトレイル・ポインタの値は共に 0 となっ

ている。

【 0 0 4 5 】

図 5 は、本実施例の計算機システムにおける、I B 5 の各々のエントリ 5 (i) の詳細な構成を示す説明図である。ここで、i はエントリのアドレスである。I B 5 の各々のエントリ 5 (i) はオペレーション・フィールド 5 0 (i)、オペランド・フィールド 5 1 (i)、第 1 ソース・フィールド 5 2 (i)、第 2 ソース・フィールド 5 3 (i)、第 3 ソース・フィールド 5 4 (i)、第 4 ソース・フィールド 5 5 (i)、第 1 デスティネーション・フィールド 5 6 (i)、第 2 デスティネーション・フィールド 5 7 (i)、分岐タグ (B T) フィールド 5 8 (i)、及び実行状態 (S ; State) フィールド 5 9 (i) から成っている。

【 0 0 4 6 】

I B の各々のエントリのオペレーション・フィールドはオペレーション・コードが書き込まれる構成となっている。

【 0 0 4 7 】

I B の各々のエントリのオペランド・フィールドは、オペレーション・コードに続いてオペランドが示されるような命令の場合に、このオペランドが書き込まれるようになっている。

【 0 0 4 8 】

I B の各々のエントリの第 1 ～第 4 ソース・フィールドの各々は、ソース・データを保持すべく割り付けられている C R F のエントリのアドレスが書き込まれるようになっている。ポップ操作を含む命令がデコードされた場合には、ポップすべき語数と同じ数だけ A P S からポップされる C R F のエントリのアドレスが、その順で第 1 ～第 4 ソース・フィールドに書き込まれるようになっている。

【 0 0 4 9 】

I B の各々のエントリの第 1 ～第 2 デスティネーション・フィールドの各々は、命令のデコード・発行に伴い、新たに割り付けられる C R F のエントリのアドレスが書き込まれるようになっている。プッシュ操作を含む命令がデコードされた場合には、プッシュすべき語数と同じ数だけ A P S にプッシュされる C R F のエントリのアドレスが、その順で第 1 ～第 2 デスティネーション・フィールドに

書き込まれるようになっている。

【 0 0 5 0 】

I B の各々のエントリの B T フィールドは、分岐予測に基づく投機的実行に係るもので、本実施例においては、後述するように、B T フィールドに書き込まれる分岐タグは A P S 履歴ファイルのエントリのアドレスと一定の関係にある。

【 0 0 5 1 】

I B の各々のエントリにおいて、S フィールドは、そのエントリに書き込まれている命令の実行状態に応じて、未実行、実行済み、正常終了、例外事象発生等の情報が書き込まれているようになっている。

【 0 0 5 2 】

(G) 共通データ・バス (C D B ; Common Data Bus)

共通データ・バス (以下では C D B と記す) は、後述する機能ユニット群及び C R F の間で、データを C R F のエントリのアドレスと共に分配する通信チャンネルである。C D B は、十分なデータ通信バンド幅が確保できるように、多重化されている。

【 0 0 5 3 】

(H) 機能ユニット

本実施例の計算機システムは、演算ユニット 0 及び 1、分岐ユニット及びロード/ストア・ユニットの 4 つの機能ユニットを具備する。本実施例においては、各機能ユニットは、基本的に、2 つのリザーベーション・ステーションと割り当てられた命令を処理する実行部で構成される。リザーベーション・ステーション (以下では R S と記す) は、命令の内容を一時的に保持するバッファであるが、本実施例の計算機システムにおいては、命令のデコードの際に、同じ命令の内容が書き込まれる I B のエントリのアドレスも書き込まれるような構成となっている。

【 0 0 5 4 】

命令のデコードの際に、命令の種類に応じて、必要な場合に、適切な機能ユニットの空いている R S に書き込みが行われるようになっている。

【 0 0 5 5 】

各機能ユニットの各々の R S で、そこに書き込まれているソース・データを保

持すべき C R F のエントリのアドレスと C D B で送られてくるものが比較され、一致すればデータが取り込まれるようになっている。

【0056】

命令の内容を保持している R S において、必要なソース・データが揃い、機能ユニットの実行部が利用可能であれば、当該 R S の内容は実行部に渡され、その実行が開始されるようになっている。

【0057】

(H-1) 演算ユニット

本実施例の計算機システムは、演算ユニット 0 及び演算ユニット 1 を具備しており、その各々の実行部は算術論理演算、データ・タイプの変換演算、比較演算等の演算命令を実行する機能を有し、互いに独立に並行して動作することができるようになっている。

【0058】

本発明の計算機システムにおいては、各々の演算ユニットの実行部をパイプライン化したり、より多くの演算ユニットを具備したり、演算ユニットごとに実行する演算の種類を特定した構成とすることも可能である。

【0059】

(H-2) 分岐ユニット

分岐ユニットの実行部は、条件分岐命令を処理し、分岐の有無を確定して、分岐先アドレスと共に、命令フェッチ・ユニットに通知する機能を有する。

【0060】

(H-3) ロード／ストア・ユニット (L S U ; Load/Store Unit)

及びデータ・バッファ

ロード／ストア・ユニット（以下では L S U と記す）の実行部は、アドレス計算を行う機能を有し、データ・バッファ及びデータ・キャッシュにアクセスすることができるようになっている。

【0061】

データ・バッファは、各々のエントリに 1 語のデータが書き込まれるようになっている循環型のバッファである。本発明の計算機システムにおいては、ワード

・スタックの最上位の部分がポインタ・スタックとCRFによって保持されるが、その下の部分がデータ・バッファ、さらにその下の部分がデータ・キャッシュに格納されるようになっている。LSUはデータ・バッファに高速にアクセスできるので、アクセスすべき変数データがデータ・バッファに保持されている割合が大きいほど、より効率的な計算が可能となる。また、データ・バッファに適当な語数のデータを溜めておくようにすることによって、後述するCRF-データ・バッファ-データ・キャッシュの間のSpill/Fillの動作を効率的に行うことができる。

【0062】

LSUは、最初のローカル変数へのポインタを保持する図示していないレジスタ（varsレジスタ）を具備する。本実施例の計算機システムにおいては、最初のローカル変数の格納域はデータ・バッファあるいはデータ・キャッシュにあるが、varsレジスタには、データ・キャッシュにおける相当するアドレス値が書き込まれているようになっている。すなわち、全てあるいは一部のローカル変数のデータが実際にはデータ・バッファに保持されていても、各々のローカル変数に、全てのローカル変数をデータ・キャッシュにSpillしたと仮定した場合のデータ・キャッシュにおけるアドレス値を対応させることができるので、ロード／ストア命令の処理において、LSUはvarsレジスタの値を用いてアドレス計算を行い、対象となるローカル変数の格納域がデータ・バッファかデータ・キャッシュかを判定し、その格納域にアクセスする。

【0063】

LSUは、先行命令が全て完了するまでストア命令をプログラム上の順番で保持する、図示していないストア・バッファを具備する。即ち、ストア命令は全ての先行命令が完了してから実行されるようになっている。ストア・バッファは連想機能を備えており、LSUは先行ストア命令に対する依存性の検証を行い、ロード命令の実行をout-of-orderで行うことができるようになっている。

【0064】

即ち、ロード・アドレスが先行ストア命令のストア・アドレスに一致するか、あるいは、先行ストア命令のストア・アドレスが未計算の場合（この場合、依存関

係の検証はできないので、依存関係は存在するとみなす)、当該ロード命令は先行ストア命令に対して依存関係を持つことになる。依存関係が全く存在しない場合、データ・バッファ/データ・キャッシュから直ちにデータをロードする。ロード命令が先行ストア命令に対して依存関係にあると、データ・バッファ/データ・キャッシュは正しい値を持っていないので、データ・バッファ/データ・キャッシュからデータをロードすることはできない。ロード・アドレスが先行ストア命令のストア・アドレスと一致し、ストア・データが有効であれば、そのストア命令の完了を待たずに、ストア・バッファから直接データをロードする。

【0065】

LSUは、プログラム中に示されるロード/ストア命令を実行すると共に、オーバーフロー/アンダーフローの回避のため、あるいは、メソッドの呼び出し/実行終了に伴いワード・スタックの最上位においてフレームが生成/破棄されるのに対応して、CRFに保持されているスタックの最下位にあたるデータをデータ・バッファとの間で自動的にSpill/Fillするようになっている。(ちなみに、メソッドの呼び出しにおいては、varsレジスタの値を変更した上で、スタック・トップからのストア命令も併用するのが望ましい。)

【0066】

1語分のデータをCRFからデータ・バッファにSpillするには、APSとCPSで、(BP_OF_PSで示される)CRFのエントリのアドレスを保持する最下位のエントリの内容が一致していなくてはならない(そうでない場合は一致するまで待つ)。その場合、その一致する内容で示されるCRFのエントリに書き込まれている1語分のデータをデータ・バッファにSpillすることができる。その際、BP_OF_PSの値に1を加え、上記CRFのエントリのBBフィールドを0に変更し、そのエントリのアドレスをFLに登録する。

【0067】

逆に、データ・バッファからCRFに1語分のデータをFillするには、データ・バッファからFillすべき1語分のデータを取り出し、それに空いているCRFの1エントリを割り付け、そのデータ・フィールドに書き込む。WCF、BBの各フィールドは1とする。さらに、その割り付けられたCRFのエントリのアドレ

スを、APS及びCPSの(BP_OF_PSで示される)CRFのエントリのアドレスを保持する最下位のエントリの1つ下に各々書き込み、BP_OF_PSの値から1を引く。

【0068】

データ・バッファとデータ・キャッシュの間でも、データ・バッファの空きに応じて適宜Spill/Fillの動作が行われるようになっている。

【0069】

CRF-データ・バッファ-データ・キャッシュの間で一度に複数語のデータをSpill/Fillできるようにするには、APSとCPSの2つのポインタ・スタック、データ・バッファ及びデータ・キャッシュをインタリーブ分割して、対応する分割部分間で上述と同様な動作を行うような構成とすればよい。この場合、Spill/Fillのために、APSとCPSの2つのポインタ・スタック、データ・バッファ及びデータ・キャッシュにおいて、バンク毎に1つのread/writeポート、さらに、CRFにおいてインタリーブ分割の数だけのread/writeポートが必要となる。

【0070】

ついで、本発明実施例の計算機システムの動作を説明する。

【0071】

本実施例の計算機システムは命令を、①命令フェッチ、②命令デコード・発行、③実行、④完了の4ステージで処理する。当分の間、説明を簡単にするため、1サイクルで1つの命令をデコード・発行／完了できるものとして、以下に各ステージごとに動作内容を説明する。

【0072】

① 命令フェッチ・ステージ

このステージでは、命令フェッチ・ユニットが命令キャッシュから命令を取り出すと共に、次にフェッチする命令のアドレスを決定する。次に命令をフェッチするのは通常次アドレス値からであるが、フェッチした命令が無条件分岐命令であるか、条件分岐命令で分岐すると予測した場合、分岐予測が外れた場合、あるいは例外が発生した場合には、フェッチするアドレス値を変更する。

【0073】

② 命令デコード・発行ステージ

このステージでは、命令をデコードして、命令の内容に応じて前進ポインタ・スタック（A P S）及び統合レジスタ・ファイル（C R F）を操作すると共に、命令の内容を命令バッファ（I B）及び、必要な場合には、適切な機能ユニットの空いている R S に書き込むことにより、プログラムに含まれる命令が out-of-order で実行されるべく設定する。以下に、設定動作を詳細に説明する。

【0074】

本発明の計算機システムにおいては、従来のスタック・マシンにおけるワード・スタックのスタックトップ近傍がポインタ・スタックと C R F によって再現されるが、命令に規定されているオペランド・スタックに対するスタック操作が、A P S に対して同様に適用される。ここで、1 語のデータのオペランド・スタックへのプッシュ操作をエミュレートするには、そのデータを保持すべく空いている C R F の 1 エントリを割り付け、そのエントリのアドレスを A P S にプッシュすればよい。

【0075】

即ち、デコードされた命令においてオペランド・スタックに対するポップ操作が規定されている場合には、ポップすべき語数と同じ数だけ C R F のエントリのアドレスを A P S からポップする。デコードされた命令においてオペランド・スタックに対するプッシュ操作が規定されている場合には、プッシュすべき語数と同じ数だけ空いている C R F のエントリを割り付け、上記割り付けた C R F のエントリのアドレスを A P S にプッシュする。

【0076】

オペランド・スタックの操作命令（Java VM における pop, pop2, dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2, swap）の場合、基本的には、オペランド・スタックに対して行うべき操作を A P S に対して同様に行えばよい。本実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令（Java VM における dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2）の場合には、コピー・データを保持すべく空いている C R F のエントリを割り付け、そのエントリのアドレスを A P S の適切なエントリに書き込むようになっている。

【0077】

命令のデコード・発行に伴い新たに割り付けられるCRFのエントリにおいては、BBフィールドに1を立て、Cフィールドには命令デコード・発行ユニットから送られてくる分岐タグを書き込む。即値データのプッシュ命令の場合には、データがすでに得られているので、データ・フィールドにそのデータを書き込み、WCFフィールドに1を立てる。それ以外の場合には、データはデコード・発行の時点では得られていないので、WCFフィールドを0としておく。

【0078】

デコードされた命令の内容をプログラム上の順番でIBに保持しておくために、その命令の内容をIBのトレイル・ポインタで示されるエントリに書き込み、トレイル・ポインタの値に1を加える。すなわち、オペレーション・フィールドにオペレーション・コードを書き込み、オペレーション・コードに続いてオペランドが示されるような命令の場合には、オペランド・フィールドにこのオペランドを書き込む。BTフィールドには命令デコード・発行ユニットから送られてくる分岐タグを書き込む。Sフィールドに関しては、無条件分岐命令、即値データのオペランド・スタックへのプッシュ命令あるいはスタック上でコピーを作成することのないオペランド・スタックの操作命令（Java VMにおけるpop, pop2, swap）の場合は実行済みとし、その他の命令の場合は未実行としておく。

【0079】

ポップ操作を含む命令の場合には、ポップすべき語数と同じ数だけAPSからポップされるCRFのエントリのアドレスを、その順で第1～第4ソース・フィールドに書き込む。プッシュ操作を含む命令の場合には、プッシュすべき語数と同じ数だけAPSにプッシュされるCRFのエントリのアドレスを、その順で第1～第2デスティネーション・フィールドに書き込む。

【0080】

本実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令の場合には、コピー元となるデータを保持すべく割り付けられているCRFのエントリのアドレスをソース・フィールドに、コピー・データを保持

すべく新たに割り付けられるCRFのエントリのアドレスをデスティネーション・フィールドに、一定の対応関係のもとに書き込む。

【0081】

命令の種類に応じて、オペランド・スタックに対してポップ／プッシュすべき語数（オペランド・スタックの操作命令の場合には、作成するコピーの語数）は決まっているので、オペレーション・フィールドの内容によって、第1～第4ソース・フィールド及び第1～第2デスティネーション・フィールドのうちのいずれが有効であるかを知ることができる。

【0082】

命令の内容を、IBに書き込むと同時に、命令の種類に応じて、必要な場合に、適切な機能ユニットの空いているRSにも、書き込みの行われるIBのエントリのアドレス（ここでは、1サイクル当たり1命令までしか発行しないとしているので、トレイル・ポインタの値と一致する）と共に書き込む。ここで、RSへの書き込みが必要でないのは、即値データのオペランド・スタックへのプッシュ命令、スタック上でコピーを作成することのないオペランド・スタックの操作命令及び無条件分岐命令の場合である。本実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令の場合には、その内容を演算ユニット1の空いているRSに書き込むことにする。

【0083】

エントリ・アドレスがIBのソース・フィールドに書き込まれる（APSからポップされる）CRFのエントリの各々のWCFフィールド及びデータ・フィールドが読み出され、WCFが1の場合、次サイクル以降にエントリ・アドレスとデータがCDBに載せられる。

【0084】

③ 実行ステージ

命令デコード・発行ステージにおいてあるRSに書き込まれた命令に関して、原則として以下のような動作が順次行われる。

【0085】

・各々のRSで、そこに書き込まれているソース・データを保持すべきCRFの

エントリのアドレスとCDBを通じて送られてくるものが比較され、一致すればデータが取り込まれる。本実施例においては、RSに命令の内容が書き込まれるのと同じタイミングでCDBを通じて送られてくるデータも当該RSに取り込まれるものとする。

【0086】

・必要なソース・データが揃い、機能ユニットの実行部が利用可能であれば、当該RSの内容は実行部に渡され、その実行が開始される。この時点で、当該RSにおける当該命令の保持が解除される。

【0087】

・デコードの際IBのデスティネーション・フィールドに書き込みが行われる（APSにCRFのエントリのアドレスがプッシュされる）ような命令の場合、命令の実行の結果得られたデータをデスティネーションであるCRFのエントリのアドレスと共にCDBに載せる。CRFにおいては、CDBを通じて送られてくる内容に基づきデータの書き込みを行い、WCFフィールドを1に変更する。

【0088】

・以上のような動作が全て正常に終了すれば、当該RSに書き込まれていたエントリ・アドレスにある（当該命令を保持している）IBのエントリのSフィールドを正常終了に変更する。

【0089】

以上は、大部分の命令について当てはまる原則的な動作であるが、本実施例の計算機システムにおいては、命令の種類によっては、以下のような例外的な動作が行われる。

【0090】

・LSUのRSにおいて、オペランド・スタック上のデータをポップしてアドレス計算を行うようなストア命令（Java VMにおける iastore, lastore, fastore, dastore, aastore, bastore, castore, sastore）が書き込まれている場合には、ソース・データが全て揃っていなくても、アドレス計算に必要なソース・データが揃った時点でストア・アドレスを計算し、ストア・バッファに書き込む。

【0091】

・LSUのRSにおいて、ストア命令が書き込まれている場合には、ストア・アドレスとストア・データのストア・バッファへの書き込みが共に終了すれば、当該RSに書き込まれていたエン트리・アドレスにある（当該ストア命令を保持している）IBのエントリのSフィールドをストア実行可能に変更する。前述したように、実際のストアの実行は完了ステージにおいて行う。

【0092】

・演算ユニット1のRSにおいて、スタック上でコピーを作成するようなオペランド・スタックの操作命令が書き込まれている場合には、ソース・データが書き込まれると、そのデータを、デスティネーションとして一定の対応関係のもとに書き込まれているCRFのエントリのアドレスと共にCDBに載せる。それぞれのデスティネーションに関するデータ転送が全て正常に終了すれば、当該RSに書き込まれていたエン트리・アドレスにある（当該命令を保持している）IBのエントリのSフィールドを正常終了に変更する。

【0093】

以上のように、IBに保持されている未実行の命令は、データ駆動の原理に基づき、実行可能となったものから処理されるので、命令実行順序はout-of-orderになる。また、演算ユニット0/1、分岐ユニット及びロード/ストア・ユニットの各機能ユニットは互いに独立に並行して動作する。

【0094】

ある命令の処理において例外事象が発生した場合には、その情報を、その命令を保持しているIBのエントリのSフィールドに書き込むと共に、命令フェッチ・ユニットに例外ベクタを通知する。

【0095】

④ 完了ステージ

ある命令が完了できるためには、プログラム上の順番でその命令よりも前にある命令が全て完了していなくてはならない。

【0096】

IBのヘッダ・ポインタで示されるエントリにおいて、Sフィールドが実行済み/正常終了である、あるいはそうなると、そのエントリに書き込まれている命

令の内容に基づいてC P S及びC R Fを操作し、ヘッダ・ポインタの値に 1 を加える。

【 0 0 9 7 】

C P Sは、命令がデコード・発行された際のA P Sの動作を再現すべく操作される。すなわち、ポップ／プッシュ操作を含む命令の場合には、有効なソース・フィールドの内容と同じものを順にC P Sからポップし、有効なデスティネーション・フィールドの内容を順にC P Sにプッシュする。スタック上でコピーを作成することのないオペランド・スタックの操作命令の場合には、オペランド・スタックに対して行うべき操作をC P Sに対して全く同様に行えばよい。本実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令の場合には、有効なソース・フィールド及び有効なデスティネーション・フィールドを参照して、その命令のデコード・発行の際にA P Sに対して行われた操作がC P Sにおいて再現される。

【 0 0 9 8 】

本実施例においては、上述のC P Sに対する操作に伴い、エントリ・アドレスがC P SからポップされるC R Fのエントリでは、B Bフィールドを 0 に変更し、そのエントリ・アドレスをF Lに登録する。

【 0 0 9 9 】

I Bのヘッダ・ポインタで示されるエントリにおいて、ストア命令が書き込まれている場合には、Sフィールドがストア実行可能である、あるいはそうなり、L S Uに実際のストアの実行を依頼する。こうすれば、データがプログラム上の順番でストアされることが保証できる。さらに、C P S及びC R Fに対する操作を上と同様に行い、ヘッダ・ポインタの値に 1 を加える。

【 0 1 0 0 】

以上のように、ヘッダ・ポインタの値に 1 が加えられることによって、キューから除外されたI Bのエントリに保持されていた命令は、完了したことになる。その命令よりも前に発行された命令はすべて完了しているので、命令の完了はin-orderで行われることになる。

【 0 1 0 1 】

IBのヘッダ・ポインタで示されるエントリにおいて、Sフィールドが例外事象発生である、あるいはそうなった場合には、その時点におけるCPS及びCRFによって、プログラムがin-orderで実行された場合の例外発生時点の状態が構成されるので、正確な例外処理が可能である。例外事象の発生した命令以降に発行された命令を全てキャンセルするには、キャンセルされるべき命令が書き込まれているIBのエントリの有効なデスティネーション・フィールドに示されるCRFのエントリの各々に対して、そのBBフィールドを0に戻し、そのエントリ・アドレスをFLに登録することによって、割り付けを解除し、ヘッダ・ポインタの値に1を加えたものをトレイル・ポインタに書き込むことによって、キャンセルされるべき命令を保持しているIBのエントリを全てキューから除外すればよい。

【0102】

以上が、本発明実施例の計算機システムの全般的な動作である。

【0103】

ついで、具体的な動作例について説明する。いま、本実施例の計算機システムで、以下のようなプログラムを実行することを考えよう。

dload [A] (変数名[A]に対応する倍精度浮動小数点データのロード)
 dload [B] (変数名[B]に対応する倍精度浮動小数点データのロード)
 dadd (倍精度浮動小数点データ間の加算)
 d2f (倍精度浮動小数点データの単精度浮動小数点データへの変換)
 fload [T] (変数名[T]に対応する単精度浮動小数点データのロード)
 swap (スタック上の最上位の2語を入れ替える)
 dup_x1 (スタックトップの語のコピーを作成し、先頭から3語目に割り込ませる)
 fsub (単精度浮動小数点データ間の減算)
 fdiv (単精度浮動小数点データ間の除算)
 fstore [X] (スタックトップにある単精度浮動小数点データの変数名[X]に対応する格納域へのストア)

【0104】

以上のプログラムは、 $X=(A+B)/(T-(A+B))$ の計算を行うものであるが、AとBのデータが倍精度で与えられ、この間の加算を倍精度のまま実行して、得られた加算データを単精度に変換して、以降は単精度で計算を行う、というものである。

【0105】

図6～図14は、本実施例の計算機システムにおいて、上記プログラムを処理する際の動作をサイクル毎に示した説明図であり、以下ではこの図をもとに詳細な動作を説明する。図6～図14において、CRF6及びIB5の各エントリの構成は、それぞれ図3、図5のものと同じである。図6～図14で空白となっている箇所は、そのフィールドの内容に留意する必要が無いことを意味する。時系列で各構成要素の内容を示すために、各部の符号の後尾にハイフンと各サイクルに対応する数字を添えている。また、図6～図14において、APS、CPS、IB及びCRFの各エントリは下から順に0、1、2、～のようにアドレスが付けられているものとする。

【0106】

CDBは3本のバスで構成されているものとする。レイテンシが2サイクル以下の演算命令は演算ユニット0で、それ以外の演算命令は演算ユニット1で実行されるものとする。

【0107】

本動作例においては、説明を簡単にするため、変数データは全てデータ・バッファに保持されており、CRFとデータ・バッファの間のSpill/Fillの動作は行わないものとする。従って、BP_OF_PS の値は終始0である。

【0108】

また、本動作例においては、当初、APS、CPS、IB及びCRFは初期化されており、FLにCRFの全てのエントリのアドレスが順に<0>,<1>,<2>,<3>……と書き込まれていて、この順で取り出されるものとする。

【0109】

以下に、各サイクルにおける動作を、(A) 命令デコード・発行、(B) 実行及び(C) 完了の各ステージに分けて詳細に説明する。

【0110】

(1-A) 第1サイクルの命令デコード・発行ステージ

命令dload [A]のデコード・発行を行う。倍長語の変数データのオペランド・スタックへのロード命令であるので、FLに登録されているフリーなCRFの2エントリ6(0)、6(1)をそのデータを保持すべく割り付け、そのエントリのアドレス<0>、<1>をAPSにプッシュし、APSは3-1のようになる。

【0111】

CRFの6(0)、6(1)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込み、CRFは6-1のようになる。ここで、本動作例においては、終始分岐タグとして命令デコード・発行ユニットから0が送られてくるものとする。

【0112】

トレイル・ポインタの値は0であるので、IBのエントリ5(0)に上記命令の内容を書き込み、IBは5-1のようになる。この際、APSにプッシュされるCRFのエントリのアドレス<0>、<1>を各々第1、第2デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え1にする。ここで、本動作例においては、IBのSフィールドには、命令が未実行であれば0、実行済み／正常終了あるいはストア命令におけるストア実行可能であれば1が書き込まれるものとする。

【0113】

IBのエントリ5(0)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス0と共に、LSUの空いているRS831に書き込む。

【0114】

(1-B) 第1サイクルの実行ステージ

実行ステージの動作としては何も行われぬ。

【0115】

(1-C) 第1サイクルの完了ステージ

当初のIBのヘッダ・ポインタが示すエントリ5(0)において、命令はまだ書き込まれていないため、完了ステージの動作としては何も行われぬ。

【0116】

(2-A) 第2サイクルの命令デコード・発行ステージ

命令dload [B]のデコード・発行を行う。倍長語の変数データのオペランド・スタックへのロード命令であるので、FLに登録されているフリーなCRFの2エントリ6(2)、6(3)をそのデータを保持すべく割り付け、そのエントリのアドレス<2>、<3>をAPSにプッシュし、APSは3-2のようになる。

【0117】

CRFの6(2)、6(3)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込み、CRFは6-2のようになる。

【0118】

トレイル・ポインタの値は1であるので、IBのエントリ5(1)に上記命令の内容を書き込み、IBは5-2のようになる。この際、APSにプッシュされるCRFのエントリのアドレス<2>、<3>を各々第1、第2デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え2にする。

【0119】

IBのエントリ5(1)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス1と共に、LSUの空いているRS832に書き込む。

【0120】

(2-B) 第2サイクルの実行ステージ

LSUの実行部はRS831から渡されるロード命令を実行する。即ち、データ・バッファにアクセスし、変数Aの2語のデータを読み出す。

【0121】

(2-C) 第2サイクルの完了ステージ

5-1の状態にあるIBのヘッダ・ポインタが示すエントリ5(0)において、Sフィールドは0であるので、完了ステージの動作としては何も行われぬ。

【0122】

(3-A) 第3サイクルの命令デコード・発行ステージ

命令daddのデコード・発行を行う。オペランド・スタックから4語のソース・デ

ータをポップして演算を行い、倍長語の演算結果をプッシュする演算命令であるので、APSから〈0〉, 〈1〉, 〈2〉, 〈3〉をポップし、FLに登録されているフリーなCRFの2エントリ6(4)、6(5)を演算結果を保持すべく割り付け、そのエントリのアドレス〈4〉, 〈5〉をAPSにプッシュし、APSは3-3のようになる。

【0 1 2 3】

CRFの6(4)、6(5)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

【0 1 2 4】

トレイル・ポインタの値は2であるので、IBのエントリ5(2)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス〈0〉, 〈1〉, 〈2〉, 〈3〉を各々第1～第4ソース・フィールドに、APSにプッシュされる〈4〉, 〈5〉を各々第1、第2デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え3にする。

【0 1 2 5】

IBのエントリ5(2)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス2と共に、演算ユニット0の空いているRS801に書き込む(daddの演算のレイテンシは2サイクルであるとする)。

【0 1 2 6】

また、6-2の状態にあるCRFの6(0)、6(1)、6(2)、6(3)の各エントリのWCFフィールド及びデータ・フィールドが読み出され、この場合、いずれのエントリもWCFが0であるので、データ転送の必要はない。

【0 1 2 7】

(3-B) 第3サイクルの実行ステージ

LSUは、データ・バッファから読み出した変数Aのデータを構成する2語A₁、A₂を、それぞれデスティネーションであるCRFのエントリのアドレス〈0〉, 〈1〉と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ6(0)、6(1)にデータの書き込みを行い、WCFフィールドを1に変更する。また、同じタイミングでIBのエントリ5(2)に書き込まれるものと同じ内容が

書き込まれる演算ユニット 0 の R S 8 0 1 においても、C R F のエントリのアドレス < 0 > , < 1 > に対応するデータの書き込みが行われる。

【 0 1 2 8 】

以上で、I B のエントリ 5 (0) に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5 (0) の S フィールドが正常終了を意味する 1 に変更される。

【 0 1 2 9 】

以上の動作と並行して、L S U の実行部は R S 8 3 2 から渡されるロード命令を実行する。即ち、データ・バッファにアクセスし、変数 B の 2 語のデータを読み出す。

【 0 1 3 0 】

(3 - C) 第 3 サイクルの完了ステージ

5 - 2 の状態にある I B のヘッダ・ポインタが示すエントリ 5 (0) において、S フィールドは 0 であるので、完了ステージの動作としては何も行われない。

【 0 1 3 1 】

(4 - A) 第 4 サイクルの命令デコード・発行ステージ

命令 d2f のデコード・発行を行う。オペランド・スタックから 2 語のソース・データをポップして変換演算を行い、1 語の演算結果をプッシュする演算命令であるので、A P S から < 4 > , < 5 > をポップし、F L に登録されているフリーな C R F のエントリ 6 (6) を演算結果を保持すべく割り付け、そのエントリのアドレス < 6 > を A P S にプッシュし、A P S は 3 - 4 のようになる。

【 0 1 3 2 】

C R F のエントリ 6 (6) においては、B B フィールドに 1 を立て、W C F 及び C の各フィールドには 0 を書き込む。

【 0 1 3 3 】

トレイル・ポインタの値は 3 であるので、I B のエントリ 5 (3) に上記命令の内容を書き込む。この際、A P S からポップされる C R F のエントリのアドレス < 4 > , < 5 > を各々第 1、第 2 ソース・フィールドに、A P S にプッシュされる < 6 > を第 1 デスティネーション・フィールドに書き込んでいる。さらに、トレイ

ル・ポインタの値に 1 を加え 4 にする。

【0134】

IB のエントリ 5 (3) に書き込まれるものと同じ上記命令の内容を、IB のエントリのアドレス 3 と共に、演算ユニット 0 の空いている RS 802 に書き込む (d2f の演算のレイテンシは 2 サイクルであるとする)。

【0135】

また、6-3 の状態にある CRF の 6 (4)、6 (5) の各エントリの WCF フィールド及びデータ・フィールドが読み出され、この場合、いずれのエントリも WCF が 0 であるので、データ転送の必要はない。

【0136】

(4-B) 第 4 サイクルの実行ステージ

LSU は、データ・バッファから読み出した変数 B のデータを構成する 2 語 B₁、B₂ を、それぞれデスティネーションである CRF のエントリのアドレス <2>、<3> と共に、CDB に載せる。これに基づき、CRF においては、エントリ 6 (2)、6 (3) にデータの書き込みを行い、WCF フィールドを 1 に変更する。また、IB のエントリ 5 (2) に書き込まれているものと同じ内容が書き込まれている演算ユニット 0 の RS 801 においても、CRF のエントリのアドレス <2>、<3> に対応するデータの書き込みが行われる。

【0137】

以上で、IB のエントリ 5 (1) に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5 (1) の S フィールドが正常終了を意味する 1 に変更される。

【0138】

(4-C) 第 4 サイクルの完了ステージ

5-3 の状態にある IB のヘッダ・ポインタが示すエントリ 5 (0) において、S フィールドは 0 であるので、完了ステージの動作としては何も行われない。

【0139】

(5-A) 第 5 サイクルの命令デコード・発行ステージ

命令 fload [T] のデコード・発行を行う。1 語の変数データのオペランド・ス

タックへのロード命令であるので、FLに登録されているフリーなCRFのエントリ 6 (7)をそのデータを保持すべく割り付け、そのエントリのアドレス <7>をAPSにプッシュし、APSは3-5のようになる。

【0 1 4 0】

CRFのエントリ 6 (7)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

【0 1 4 1】

トレイル・ポインタの値は4であるので、IBのエントリ 5 (4)に上記命令の内容を書き込む。この際、APSにプッシュされるCRFのエントリのアドレス <7>を第1デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え5にする。

【0 1 4 2】

IBのエントリ 5 (4)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス4と共に、LSUの空いているRS 8 3 1に書き込む。

【0 1 4 3】

(5-B) 第5サイクルの実行ステージ

演算命令daddの内容が書き込まれているRS 8 0 1において、必要なソース・データが全て揃ったので、その内容が演算ユニット0の実行部に渡され、演算が開始される。

【0 1 4 4】

(5-C) 第5サイクルの完了ステージ

5-4の状態にあるIBのヘッダ・ポインタが示すエントリ 5 (0)において、Sフィールドが1となったので、5 (0)の内容に基づいてCPS (及びCRF)を操作する。すなわち、IBのエントリ 5 (0)のデスティネーション・フィールドに書き込まれている<0>,<1>をCPSにプッシュし、CPSは4-5のようになる。さらに、ヘッダ・ポインタの値に1を加え1とし、これで、5 (0)の命令は完了したことになる。

【0 1 4 5】

(6-A) 第6サイクルの命令デコード・発行ステージ

命令swapのデコード・発行を行う。オペランド・スタック上の最上位の2語を入れ替える命令であるので、同様な操作をAPSに対して行い、APSは3-6のようになる。

【0146】

トレイル・ポインタの値は5であるので、IBのエントリ5(5)に上記命令の内容を書き込む。この際、命令swapは、スタック上でコピーを作成することのないオペランド・スタックの操作命令であるので、Sフィールドは実行済みを意味する1とする。さらに、トレイル・ポインタの値に1を加え6にする。

【0147】

(6-B) 第6サイクルの実行ステージ

LSUの実行部はRS831から渡されるロード命令を実行する。即ち、データ・バッファにアクセスし、変数Tのデータを読み出す。

【0148】

(6-C) 第6サイクルの完了ステージ

5-5の状態にあるIBのヘッダ・ポインタが示すエントリ5(1)において、Sフィールドが1となったので、5(1)の内容に基づいてCPS(及びCRF)を操作する。すなわち、IBのエントリ5(1)のデスティネーション・フィールドに書き込まれている<2>,<3>をCPSにプッシュし、CPSは4-6のようになる。さらに、ヘッダ・ポインタの値に1を加え2とし、これで、5(1)の命令は完了したことになる。

【0149】

(7-A) 第7サイクルの命令デコード・発行ステージ

命令dup_x1のデコード・発行を行う。命令dup_x1は、ワード・スタックが、(右方向に成長するものとして)...., word1, word2 のような状態であるとき、これを...., word2, word1, word2 と変えるような、スタック上で1語のコピーを作成するオペランド・スタックの操作命令であるので、FLに登録されているフリーなCRFのエントリ6(8)をコピー・データを保持すべく割り付け、3-6のように下から<7>,<6>となっている状態のAPSを3-7のように<8>,<7>,<6>と変える。

【0150】

C R Fのエントリ 6 (8)においては、B Bフィールドに 1 を立て、W C F及び Cの各フィールドには 0 を書き込む。

【0151】

トレイル・ポインタの値は 6 であるので、I Bのエントリ 5 (6)に上記命令の内容を書き込む。この際、コピー元となるデータを保持すべく割り付けられている C R Fのエントリのアドレス <6> を第 1 ソース・フィールドに、コピー・データを保持すべく新たに割り付けられる C R Fのエントリのアドレス <8> を第 1 デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に 1 を加え 7 にする。

【0152】

I Bのエントリ 5 (6)に書き込まれるものと同じ上記命令の内容を、I Bのエントリのアドレス 6 と共に、演算ユニット 1 の空いている R S 8 1 1 に書き込む。

【0153】

また、6-6の状態にある C R Fのエントリ 6 (6)の W C Fフィールド及びデータ・フィールドが読み出され、この場合、W C Fが 0 であるので、データ転送の必要はない。

【0154】

(7-B) 第 7 サイクルの実行ステージ

演算ユニット 0 は、5 (2)の演算命令の実行を終了しており、演算結果を構成する 2 語 (A+B)_1、(A+B)_2 を、それぞれデスティネーションである C R Fのエントリのアドレス <4> , <5> と共に、C D Bに載せる。これに基づき、C R Fにおいては、エントリ 6 (4)、6 (5)にデータの書き込みを行い、W C Fフィールドを 1 に変更する。また、I Bのエントリ 5 (3)に書き込まれているものと同じ内容が書き込まれている演算ユニット 0 の R S 8 0 2においても、C R Fのエントリのアドレス <4> , <5> に対応するデータの書き込みが行われる。

【0155】

L S Uは、データ・バッファから読み出した変数 T のデータを、デスティネー

ションであるCRFのエントリのアドレス〈7〉と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ6(7)にデータの書き込みを行い、WCFフィールドを1に変更する。

【0156】

以上で、IBの5(2)、5(4)の各エントリに書き込まれている命令の実行が共に正常に終了するので、次のサイクルにおいて、5(2)及び5(4)のSフィールドが正常終了を意味する1に変更される。

【0157】

(7-C) 第7サイクルの完了ステージ

5-6の状態にあるIBのヘッダ・ポインタが示すエントリ5(2)において、Sフィールドは0であるので、完了ステージの動作としては何も行われない。

【0158】

(8-A) 第8サイクルの命令デコード・発行ステージ

命令fsubのデコード・発行を行う。オペランド・スタックから2語のソース・データをポップして演算を行い、1語の演算結果をプッシュする演算命令であるので、APSから〈7〉, 〈6〉をポップし、FLに登録されているフリーなCRFのエントリ6(9)を演算結果を保持すべく割り付け、そのエントリのアドレス〈9〉をAPSにプッシュし、APSは3-8のようになる。

【0159】

CRFのエントリ6(9)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

【0160】

トレイル・ポインタの値は7であるので、IBのエントリ5(7)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス〈7〉, 〈6〉を各々第1、第2ソース・フィールドに、APSにプッシュされる〈9〉を第1デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え8にする。

【0161】

IBのエントリ5(7)に書き込まれるものと同じ上記命令の内容を、IBのエ

ントリのアドレス7と共に、演算ユニット0の空いているRS801に書き込む
(fsubの演算のレイテンシは2サイクルであるとする)。

【0162】

また、6-7の状態にあるCRFの6(7)、6(6)の各エントリのWCFフィールド及びデータ・フィールドが読み出され、この場合、6(7)のWCFが1であるので、次のサイクルにおいて、エントリ・アドレス<7>とデータTがCDBに載せられる。

【0163】

(8-B) 第8サイクルの実行ステージ

演算命令d2fの内容が書き込まれているRS802において、必要なソース・データが全て揃ったので、その内容が演算ユニット0の実行部に渡され、演算が開始される。

【0164】

(8-C) 第8サイクルの完了ステージ

5-7の状態にあるIBのヘッダ・ポインタが示すエントリ5(2)において、Sフィールドは0であるので、完了ステージの動作としては何も行われぬ。

【0165】

(9-A) 第9サイクルの命令デコード・発行ステージ

命令fdivのデコード・発行を行う。オペランド・スタックから2語のソース・データをポップして演算を行い、1語の演算結果をプッシュする演算命令であるので、APSから<8>、<9>をポップし、FLに登録されているフリーなCRFのエントリ6(10)を演算結果を保持すべく割り付け、そのエントリのアドレス<10>をAPSにプッシュし、APSは3-9のようになる。

【0166】

CRFのエントリ6(10)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

【0167】

トレイル・ポインタの値は8であるので、IBのエントリ5(8)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス<

8> , <9> を各々第1、第2ソース・フィールドに、APSにプッシュされる<10> を第1デスティネーション・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え9にする。

【0168】

IBのエントリ5(8)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス8と共に、演算ユニット1の空いているRS812に書き込む(fdivの演算のレイテンシは10サイクルであるとする)。

【0169】

また、6-8の状態にあるCRFの6(8)、6(9)の各エントリのWCFフィールド及びデータ・フィールドが読み出され、この場合、いずれのエントリもWCFが0であるので、データ転送の必要はない。

【0170】

(9-B) 第9サイクルの実行ステージ

(8-A) で述べたように、エントリ・アドレス<7> とデータTがCDBに載せられ、これに基づき、IBのエントリ5(7)に書き込まれているものと同じ内容が書き込まれている演算ユニット0のRS801においても、CRFのエントリのアドレス<7> に対応するデータの書き込みが行われる。

【0171】

(9-C) 第9サイクルの完了ステージ

5-8の状態にあるIBのヘッダ・ポインタが示すエントリ5(2)において、Sフィールドが1となったので、5(2)の内容に基づいてCPS及びCRFを操作する。すなわち、IBのエントリ5(2)のソース・フィールドに書き込まれている<0> , <1> , <2> , <3> をCPSからポップし、デスティネーション・フィールドに書き込まれている<4> , <5> をCPSにプッシュし、CPSは4-9のようになる。エントリ・アドレスがCPSからポップされるCRFの6(0)、6(1)、6(2)、6(3)の各エントリでは、BBフィールドを0に変更する。CRFのエントリのアドレス<0> , <1> , <2> , <3> をFLに登録する。さらに、ヘッダ・ポインタの値に1を加え3とし、これで、5(2)の命令は完了したことになる。

【0172】

(10-A) 第10サイクルの命令デコード・発行ステージ

命令fstore [X]のデコード・発行を行う。スタックトップにある1語のデータのストア命令であるので、APSから<10>をポップし、APSは3-10のようになる。

【0173】

トレイル・ポインタの値は9であるので、IBのエントリ5(9)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス<10>を第1ソース・フィールドに書き込んでいる。さらに、トレイル・ポインタの値に1を加え10にする。

【0174】

IBのエントリ5(9)に書き込まれるものと同じ上記命令の内容を、IBのエントリのアドレス9と共に、LSUの空いているRS831に書き込む。次のサイクルにおいて、ストア・バッファに変数名[X]に対応するストア・アドレスが書き込まれる。

【0175】

また、6-9の状態にあるCRFのエントリ6(10)のWCFフィールド及びデータ・フィールドが読み出され、この場合、WCFが0であるので、データ転送の必要はない。

【0176】

(10-B) 第10サイクルの実行ステージ

演算ユニット0は、5(3)の変換演算命令の実行を終了しており、1語の演算結果(A+B)を、デスティネーションであるCRFのエントリのアドレス<6>と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ6(6)にデータの書き込みを行い、WCFフィールドを1に変更する。また、IBのエントリ5(6)、5(7)に書き込まれているものとそれぞれ同じ内容が書き込まれている演算ユニット1のRS811及び演算ユニット0のRS801においても、CRFのエントリのアドレス<6>に対応するデータの書き込みが行われる。

【0177】

以上で、I Bのエントリ 5 (3)に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5 (3)のSフィールドが正常終了を意味する1に変更される。

【0 1 7 8】

(1 0 - C) 第1 0サイクルの完了ステージ

5 - 9の状態にあるI Bのヘッダ・ポインタが示すエントリ 5 (3)において、Sフィールドは0であるので、完了ステージの動作としては何も行われぬ。

【0 1 7 9】

以下では、特に記述すべき動作内容のない場合は、実行ステージであれ完了ステージであれ項目を省くことにする。

【0 1 8 0】

(1 1 - B) 第1 1サイクルの実行ステージ

演算命令fsubの内容が書き込まれているRS 8 0 1において、必要なソース・データが全て揃ったので、その内容が演算ユニット0の実行部に渡され、演算が開始される。

【0 1 8 1】

スタック上でコピーを作成するようなオペランド・スタックの操作命令dup_x1の内容が書き込まれているRS 8 1 1において、ソース・データが書き込まれたので、そのデータ(A+B)を、対応関係にあるデスティネーションであるCRFのエントリのアドレス<8>と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ 6 (8)にデータの書き込みを行い、WCFフィールドを1に変更する。また、I Bのエントリ 5 (8)に書き込まれているものと同じ内容が書き込まれている演算ユニット1のRS 8 1 2においても、CRFのエントリのアドレス<8>に対応するデータの書き込みが行われる。

【0 1 8 2】

以上で、I Bのエントリ 5 (6)に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5 (6)のSフィールドが正常終了を意味する1に変更される。

【0 1 8 3】

(12-C) 第12サイクルの完了ステージ

5-11の状態にあるIBのヘッダ・ポインタが示すエントリ5(3)において、Sフィールドが1となったので、5(3)の内容に基づいてCPS及びCRFを操作する。すなわち、IBのエントリ5(3)のソース・フィールドに書き込まれている<4>、<5>をCPSからポップし、デスティネーション・フィールドに書き込まれている<6>をCPSにプッシュし、CPSは4-12のようになる。エントリ・アドレスがCPSからポップされるCRFの6(4)、6(5)の各エントリでは、BBフィールドを0に変更する。CRFのエントリのアドレス<4>、<5>をFLに登録する。さらに、ヘッダ・ポインタの値に1を加え4とし、これで、5(3)の命令は完了したことになる。

【0184】

(13-B) 第13サイクルの実行ステージ

演算ユニット0は、5(7)の演算命令の実行を終了しており、1語の演算結果T-(A+B)を、デスティネーションであるCRFのエントリのアドレス<9>と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ6(9)にデータの書き込みを行い、WCFフィールドを1に変更する。また、IBのエントリ5(8)に書き込まれているものと同じ内容が書き込まれている演算ユニット1のRS812においても、CRFのエントリのアドレス<9>に対応するデータの書き込みが行われる。

【0185】

以上で、IBのエントリ5(7)に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5(7)のSフィールドが正常終了を意味する1に変更される。

【0186】

(13-C) 第13サイクルの完了ステージ

5-12の状態にあるIBのヘッダ・ポインタが示すエントリ5(4)において、Sフィールドが1であるので、5(4)の内容に基づいてCPS(及びCRF)を操作する。すなわち、IBのエントリ5(4)のデスティネーション・フィールドに書き込まれている<7>をCPSにプッシュし、CPSは4-13のようになる

。さらに、ヘッダ・ポインタの値に 1 を加え 5 とし、これで、5 (4) の命令は完了したことになる。

【0187】

(14-B) 第 14 サイクルの実行ステージ

演算命令 fdiv の内容が書き込まれている RS 812 において、必要なソース・データが全て揃ったので、その内容が演算ユニット 1 の実行部に渡され、演算が開始される。

【0188】

(14-C) 第 14 サイクルの完了ステージ

5-13 の状態にある IB のヘッダ・ポインタが示すエントリ 5 (5) において、S フィールドが 1 であるので、5 (5) の内容に基づいて CPS (及び CRF) を操作する。すなわち、(6-A) における APS の動作が再現され、CPS は 4-14 のようになる。さらに、ヘッダ・ポインタの値に 1 を加え 6 とし、これで、5 (5) の命令は完了したことになる。

【0189】

(15-C) 第 15 サイクルの完了ステージ

5-14 の状態にある IB のヘッダ・ポインタが示すエントリ 5 (6) において、S フィールドが 1 であるので、5 (6) の内容に基づいて CPS (及び CRF) を操作する。すなわち、(7-A) における APS の動作が再現され、CPS は 4-15 のようになる。さらに、ヘッダ・ポインタの値に 1 を加え 7 とし、これで、5 (6) の命令は完了したことになる。

【0190】

(16-C) 第 16 サイクルの完了ステージ

5-15 の状態にある IB のヘッダ・ポインタが示すエントリ 5 (7) において、S フィールドが 1 であるので、5 (7) の内容に基づいて CPS 及び CRF を操作する。すなわち、IB のエントリ 5 (7) のソース・フィールドに書き込まれている <7> , <6> を CPS からポップし、デスティネーション・フィールドに書き込まれている <9> を CPS にプッシュし、CPS は 4-16 のようになる。エントリ・アドレスが CPS からポップされる CRF の 6 (7)、6 (6) の各エントリで

は、BBフィールドを0に変更する。CRFのエントリのアドレス<7>、<6>をFLに登録する。さらに、ヘッダ・ポインタの値に1を加え8とし、これで、5(7)の命令は完了したことになる。

【0191】

(24-B) 第24サイクルの実行ステージ

演算ユニット1は、5(8)の演算命令の実行を終了しており、1語の演算結果($A+B$)/ $\{T-(A+B)\}$ を、デスティネーションであるCRFのエントリのアドレス<10>と共に、CDBに載せる。これに基づき、CRFにおいては、エントリ6(10)にデータの書き込みを行い、WCFフィールドを1に変更する。また、IBのエントリ5(9)に書き込まれているものと同じ内容が書き込まれているLSUのRS831においても、CRFのエントリのアドレス<10>に対応するデータの書き込みが行われる。

【0192】

以上で、IBのエントリ5(8)に書き込まれている命令の実行が正常に終了するので、次のサイクルにおいて、5(8)のSフィールドが正常終了を意味する1に変更される。

【0193】

(25-B) 第25サイクルの実行ステージ

ストア命令fstoreの内容が書き込まれているRS831において、ストア・データが書き込まれたので、そのデータをストア・バッファに書き込む。

【0194】

以上で、IBのエントリ5(9)に書き込まれているストア命令に関して、ストア・アドレスとストア・データのストア・バッファへの書き込みが共に終了するので、次のサイクルにおいて、5(9)のSフィールドがストア実行可能を意味する1に変更される。

【0195】

(26-C) 第26サイクルの完了ステージ

5-25の状態にあるIBのヘッダ・ポインタが示すエントリ5(8)において、Sフィールドが1となったので、5(8)の内容に基づいてCPS及びCRFを操

作する。すなわち、IBのエントリ5(8)のソース・フィールドに書き込まれている<8>,<9>をCPSからポップし、デスティネーション・フィールドに書き込まれている<10>をCPSにプッシュし、CPSは4-26のようになる。エントリ・アドレスがCPSからポップされるCRFの6(8)、6(9)の各エントリでは、BBフィールドを0に変更する。CRFのエントリのアドレス<8>,<9>をFLに登録する。さらに、ヘッダ・ポインタの値に1を加え9とし、これで、5(8)の命令は完了したことになる。

【0196】

(27-C) 第27サイクルの完了ステージ

5-26の状態にあるIBのヘッダ・ポインタが示すエントリ5(9)においては、ストア命令が書き込まれており、Sフィールドが1となったので、LSUにデータ・バッファへのストアの実行を依頼する。さらに、5(9)の内容に基づいてCPS及びCRFを操作する。すなわち、IBのエントリ5(9)のソース・フィールドに書き込まれている<10>をCPSからポップし、CPSは4-27のようになる。エントリ・アドレスがCPSからポップされるCRFのエントリ6(10)では、BBフィールドを0に変更する。CRFのエントリのアドレス<10>をFLに登録する。さらに、ヘッダ・ポインタの値に1を加え10とし、これで、5(9)の命令は完了したことになる。

【0197】

以上で、本実施例の計算機システムにおいて $X=(A+B)/\{T-(A+B)\}$ の計算が完了したことになる。

【0198】

本発明の計算機システムにおいては、分岐予測に基づく投機的実行を実現することができる。APS履歴ファイルは、投機的実行を可能にするために具備されるものである。条件分岐命令がデコードされるごとに、APS履歴ファイルの1エントリにAPSの全エントリ及びPP_OF_APSの内容を書き込むようになっている。以下に、本実施例の計算機システムにおいて、分岐予測に基づく投機的実行がどのように行われるかについて説明する。

【0199】

前述のように、本実施例の計算機システムにおいては、命令デコード・発行ステージにおいて、命令をデコードして、命令の内容に応じて A P S 及び C R F を操作すると共に、命令の内容を I B 及び、必要な場合には、適切な機能ユニットの空いている R S に書き込むようになっている。初期状態から命令が流れ始め最初の条件分岐命令がデコードされるまでの間、発行される命令に分岐タグとして 0 を付し、この分岐タグ 0 を、命令の内容が書き込まれる I B のエントリ（と機能ユニットの R S）の B T フィールド、及び、割り付けられる C R F のエントリの C フィールドに書き込む。

【 0 2 0 0 】

最初の条件分岐命令がデコードされ分岐予測が行われる際に、分岐時点の状態を保存するために、A P S の全エントリ及び P P _ O F _ A P S の内容を A P S 履歴ファイルのアドレス 0 のエントリに書き込む。上記の分岐予測に基づいた命令の流れにおいては、分岐タグとして 1 を付し、I B （、機能ユニットの R S）及び C R F の設定を行う。

【 0 2 0 1 】

2 つ目の条件分岐命令がデコードされた時に、最初の条件分岐命令が未確定である場合、あるいは確定して予測が当たっていた場合には、A P S の全エントリ及び P P _ O F _ A P S の内容を A P S 履歴ファイルのアドレス 1 のエントリに書き込む。2 段目の分岐予測に基づいた命令の流れにおいては、分岐タグとして 2 を付し、I B （、機能ユニットの R S）及び C R F の設定を行う。

【 0 2 0 2 】

分岐予測が当たり続ければ以後同様に処理が進み、A P S 履歴ファイルへの書き込みはアドレス順に行われる。また、A P S 履歴ファイルのアドレス n のエントリに書き込みが行われてから次に書き込みが行われるまでの間に発行される命令には分岐タグとして n+1 を付すものとする。

【 0 2 0 3 】

分岐予測が外れた場合には、その条件分岐命令以降に発行された命令に付された分岐タグをもとに、各機能ユニットで実行中あるいは R S に保持されている命令をキャンセルし、C R F において C フィールドで分岐タグを照合してその一致

する全てのエントリの割り付けを解除し（BBフィールドを0に変更し、エントリ・アドレスをFLに登録する）、IBにおいてその条件分岐命令以降にキューに加えられたエントリを除外する（トレイル・ポインタの値をその条件分岐命令が書き込まれているエントリの次のアドレスに書き換える）。さらに、同じエントリ・アドレスにあるCPSのエントリとその内容が一致しないAPSの各エントリ及びPP_OF_APSに、その条件分岐命令がデコードされた際にAPS履歴ファイルに書き込まれた内容をコピーして、正しい位置の命令からデコード・発行を再開する。

【0204】

以上のように、本発明の計算機システムにおいては、APS履歴ファイルを用いることによって、条件分岐命令がデコードされ分岐予測が行われる各々の時点の状態を再構成することができるので、分岐予測に基づく投機的実行が可能である。

【0205】

以上では、説明を簡単にするため、1サイクルで同時にデコード・発行／完了できる命令は高々1つまでとして説明してきた。本発明の計算機システムにおいては、同時に複数の命令をデコード・発行／完了できる構成とすることが可能である。すなわち、FLがFIFOキューの構成となっていれば、割り付けのためにフリーな（空いている）CRFのエントリのアドレスをFLから取り出す順番は決まっており、各命令における何語ポップし何語プッシュするかというようなスタック操作の内容を把握して、同時に複数の命令をデコード・発行することができる。また、命令の完了は、基本的には、IBの書き込みの内容をもとに、命令がデコード・発行された際のAPSの動作を再現すべくCPSを操作し、エントリ・アドレスがCPSからポップされるCRFのエントリの割り付けを解除すればよいので、同時に複数の命令を完了できる構成とすることも可能である。

【0206】

同時にデコード・発行／完了できる命令の数を多くするほど、命令デコード・発行ユニットその他の制御回路が複雑になると共に、APS、CPS、IB、CRFやデータ・バッファを構成する各レジスタ・ファイルのポートの数や演算ユニッ

トの数、さらにCDBを構成するバスの数などの点で、より多量のハードウェアが必要となる。

【0207】

本発明の計算機システムにおいては、命令デコード・発行ステージの前段において、同時にデコード・発行する複数の命令の内容を統合した形式に変換するような構成とすることも可能である。あるいは、変換済のコードを命令キャッシュに蓄えておくような構成としてもよい。

【0208】

たとえば、1サイクル当り2命令までデコード・発行できるような構成をとる場合、前述の $X=(A+B)/\{T-(A+B)\}$ を計算するプログラムは図15の図表に示されるような内容に変換される。図15の図表の各段には、同時にデコード・発行される2つの命令に基づく、PP_OF_APSの増分、APSの操作内容及びIBの2エントリに書き込まれるべき命令の内容を示している。ここでは、命令発行前のAPSの内容を..... s2, s1, s0（右端がスタックトップ）、FIFOキューの構成となっているフリー・リストの内容を（取り出される順に）f1, f2, f3として記述しており、命令発行時にそれぞれ対応するCRFのエントリ・アドレスに置き換えられるようになっている。PP_OF_APSの増分の欄で示されるようにAPSのスタックトップの位置が移動するが、APSの操作内容の欄では、この移動後のスタックトップの位置が右端に対応している。また、'NC'は「変化なし(No Change)」を意味する。

【0209】

命令セットがスタック型の命令及びレジスタ型の命令を共に含むような、本発明に基づく計算機システムも実現可能である。すなわち、前進ポインタ・スタック及び完了ポインタ・スタックに加えて、各論理レジスタにそれぞれ対応して設けられた各エントリに統合レジスタ・ファイルのエントリ・アドレスが書き込まれるようになっている前進レジスタ・マッピング・テーブル及び完了レジスタ・マッピング・テーブルを具備する構成とし、スタック型の命令に関しては前進／完了ポインタ・スタックを操作し、レジスタ型の命令に関しては前進／完了レジスタ・マッピング・テーブルをアクセスするようにする。この場合、前進ポインタ・スタ

ック履歴ファイルの代わりに、各々のエントリに前進ポインタ・スタック及び前進レジスタ・マッピング・テーブル双方の内容が書き込まれるようになっている前進履歴ファイルを具備する必要がある。

【 0 2 1 0 】

【発明の効果】

以上のように、本発明の計算機システムは、正確な例外処理を保証しつつ、スタックマシンの機械語で記述されたプログラムを out-of-order で処理するものであるが、複数の機能ユニットによる並列処理やそれらのパイプライン化によって効率的な処理を行うことが可能であるという利点がある。

【 0 2 1 1 】

また、分岐予測に基づく投機的実行や、1 サイクル当り複数命令のデコード・発行／完了の可能な構成とすることにより、さらなる高速化が可能である。

【図面の簡単な説明】

【図 1】

本発明の一実施例の計算機システムの基本構成を示すブロック図である。

【図 2】

本発明の一実施例における前進ポインタ・スタック (APS) 3、完了ポインタ・スタック (CPS) 4、PP_OF_APS 90 と PP_OF_CPS 91 の2つのプッシュ・ポインタ及びボトム・ポインタ BP_OF_PS 92 の関係を示した説明図である。

【図 3】

本発明の一実施例における統合レジスタ・ファイル (CRF) 6 の各々のエントリ 6 (i) の詳細な構成を示した説明図である。

【図 4】

本発明の一実施例における命令バッファ (IB) 5 の構成を示した説明図である。

【図 5】

本発明の一実施例における命令バッファ (IB) 5 の各々のエントリ 5 (i) の詳細な構成を示した説明図である。

【図 6】

図 6～図 14 は、本発明の一実施例における一動作例の、サイクル毎の前進ポインタ・スタック (A P S) 3、完了ポインタ・スタック (C P S) 4、命令バッファ (I B) 5 及び統合レジスタ・ファイル (C R F) 6 の内容を具体的に示した説明図である。図 6 はそのうち第 1～第 3 サイクルの動作内容を示した図である。

【図 7】

図 6 に続く第 4～第 5 サイクルの動作内容を示した説明図である。

【図 8】

図 6～図 7 に続く第 6～第 7 サイクルの動作内容を示した説明図である。

【図 9】

図 6～図 8 に続く第 8～第 9 サイクルの動作内容を示した説明図である。

【図 10】

図 6～図 9 に続く第 10～第 11 サイクルの動作内容を示した説明図である。

【図 11】

図 6～図 10 に続く第 12～第 13 サイクルの動作内容を示した説明図である。

【図 12】

図 6～図 11 に続く第 14～第 15 サイクルの動作内容を示した説明図である。

【図 13】

図 6～図 12 に続く第 16～第 24 サイクルの動作内容を示した説明図である。

【図 14】

図 6～図 13 に続く第 25～第 27 サイクルの動作内容を示した説明図である。

【図 15】

本発明の計算機システムが 1 サイクル当たり 2 命令までデコードできるような構成をとる場合に、プログラムがどのように変換されるかを具体的に示す図表である。

【符号の説明】

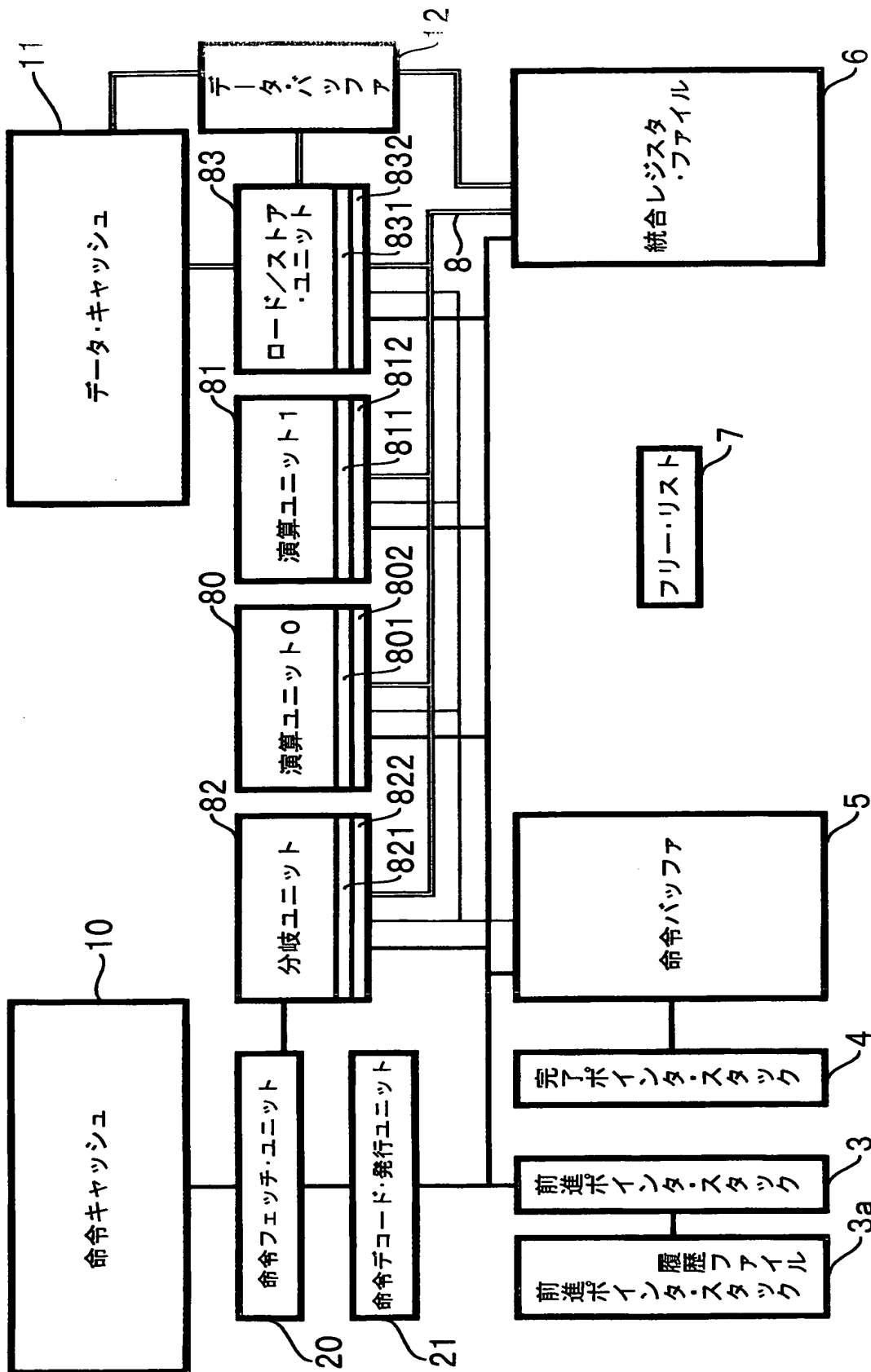
- 3 前進ポインタ・スタック (A P S)
- 3 a 前進ポインタ・スタック履歴ファイル (A P S履歴ファイル)
- 4 完了ポインタ・スタック (C P S)
- 5 命令バッファ (I B)
- 6 統合レジスタ・ファイル (C R F)
- 7 フリー・リスト (F L)
- 8 共通データ・バス (C D B)
- 10 命令キャッシュ
- 11 データ・キャッシュ
- 12 データ・バッファ
- 20 命令フェッチ・ユニット
- 21 命令デコード・発行ユニット
- 80 演算ユニット0
- 81 演算ユニット1
- 82 分岐ユニット
- 83 ロード／ストア・ユニット (L S U)
- 90 PP_OF_APS と名付けたレジスタ
- 91 PP_OF_CPS と名付けたレジスタ
- 92 BP_OF_PS と名付けたレジスタ
- 97 命令バッファ (I B) のトレイル・ポインタ
- 98 命令バッファ (I B) のヘッダ・ポインタ
- 801、802 演算ユニット0のリザーベーション・ステーション
- 811、812 演算ユニット1のリザーベーション・ステーション
- 821、822 分岐ユニットのリザーベーション・ステーション
- 831、832 ロード／ストア・ユニット (L S U) のリザーベーション・ステーション

特平 1 1 - 1 1 5 0 4 7

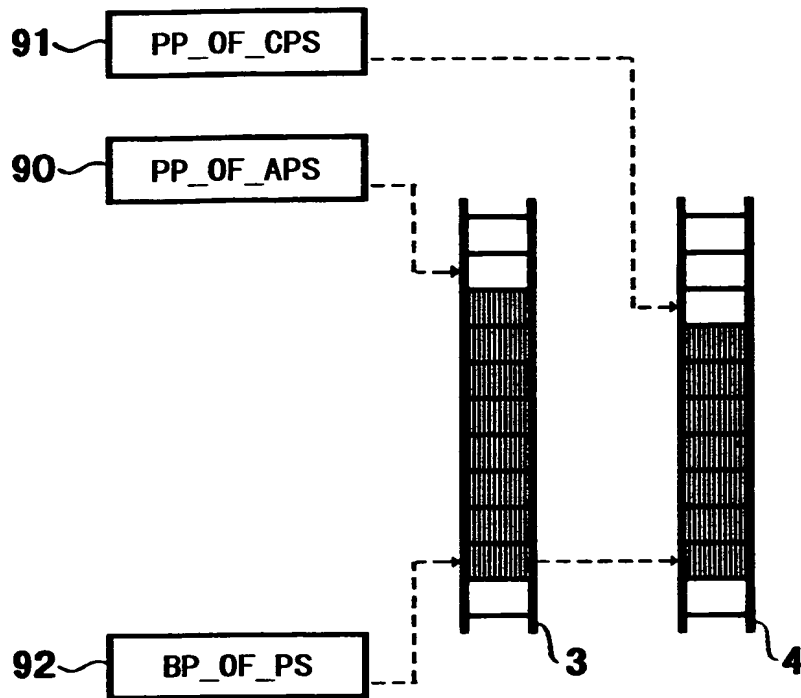
【書類名】

図面

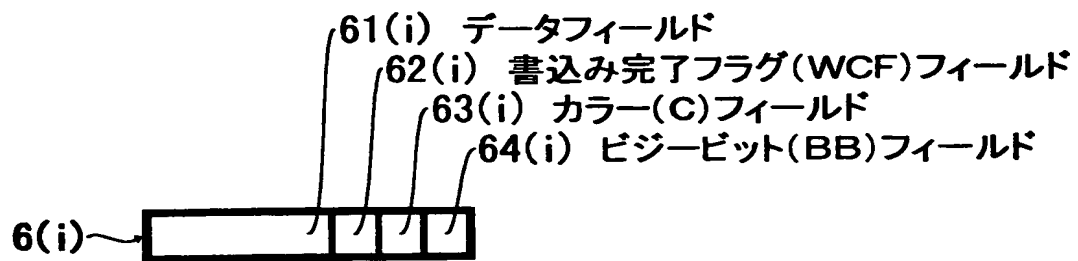
【図 1】



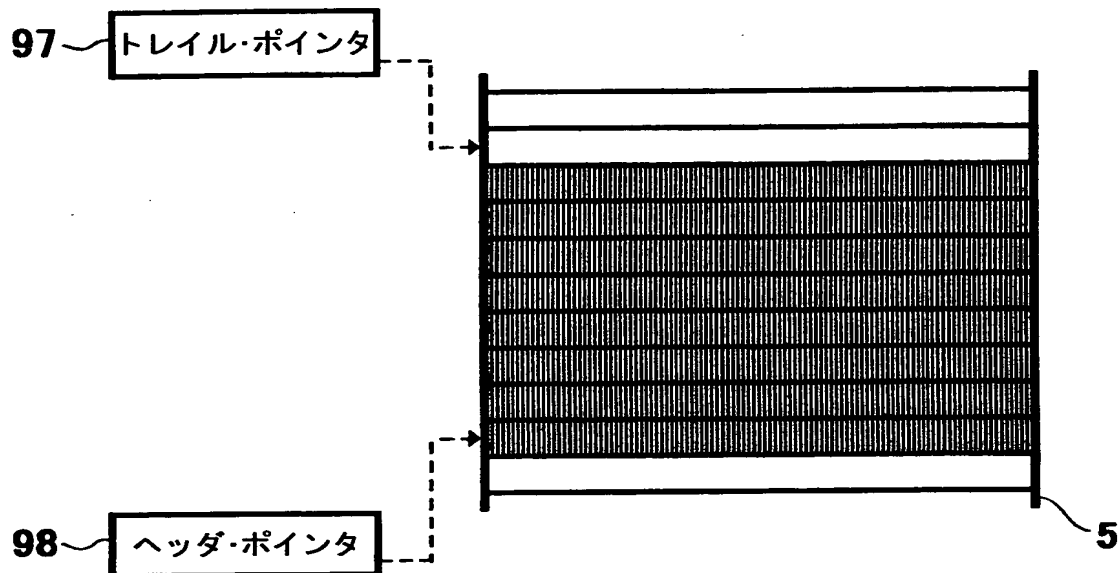
【図 2】



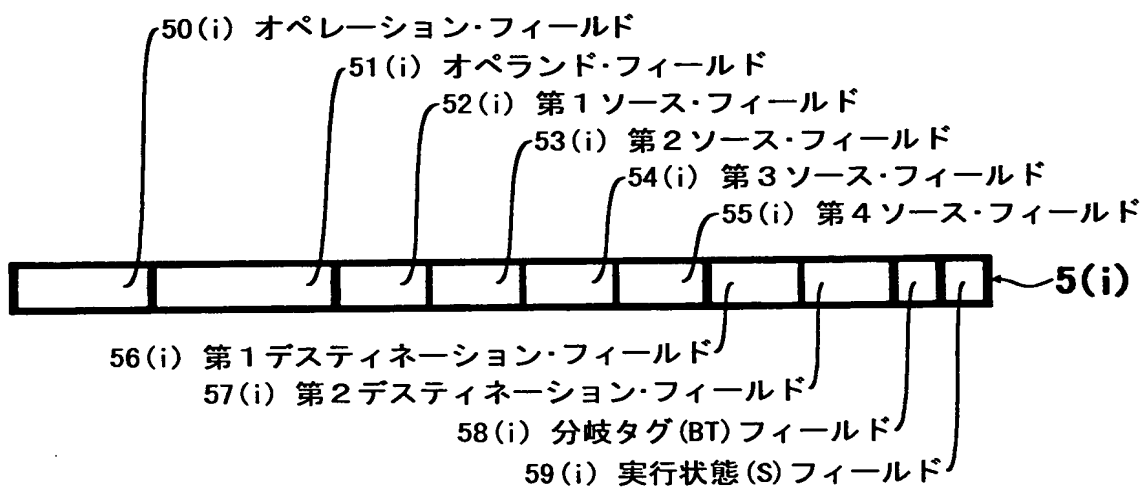
【図 3】



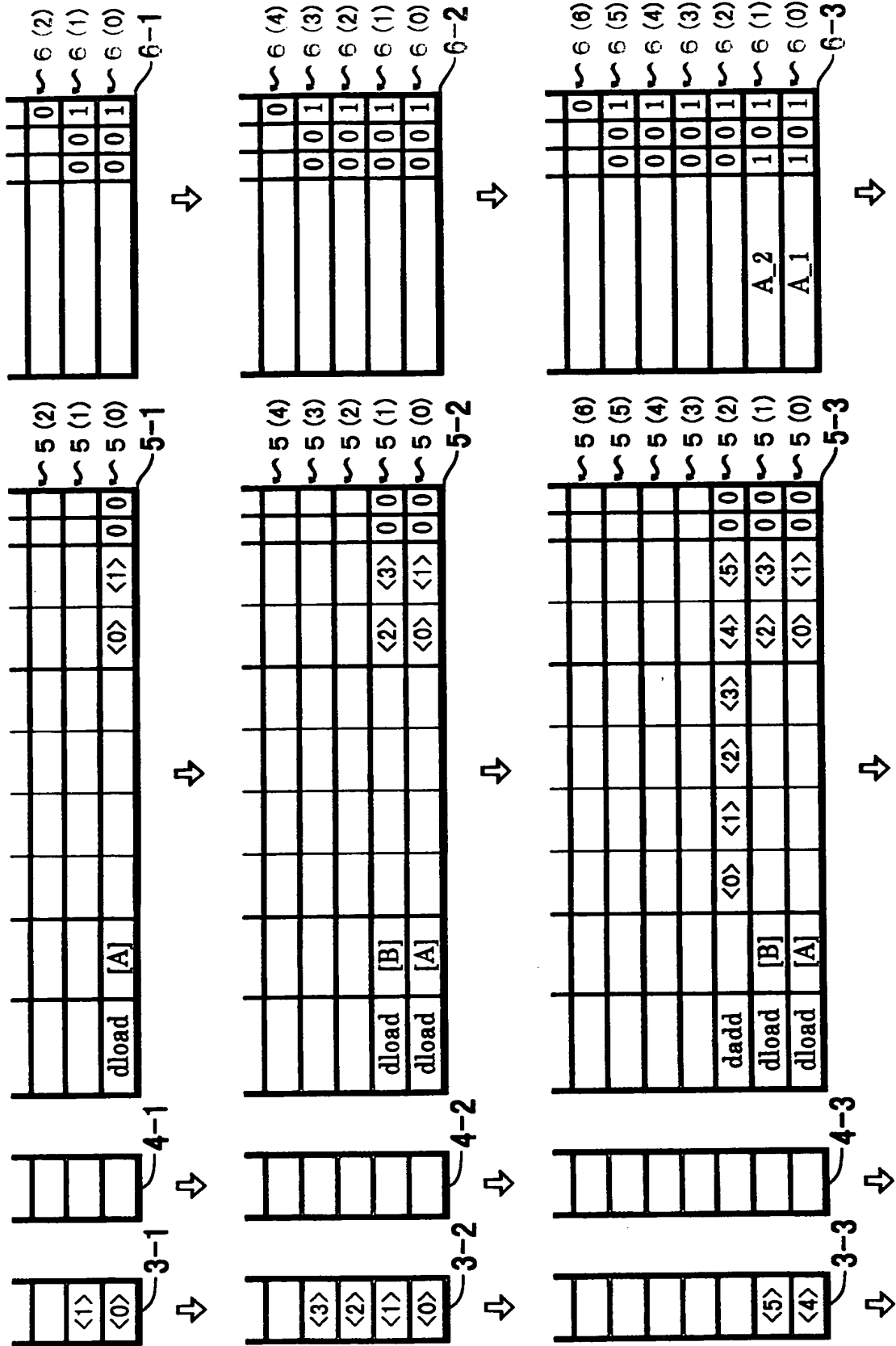
【図 4】



【図 5】

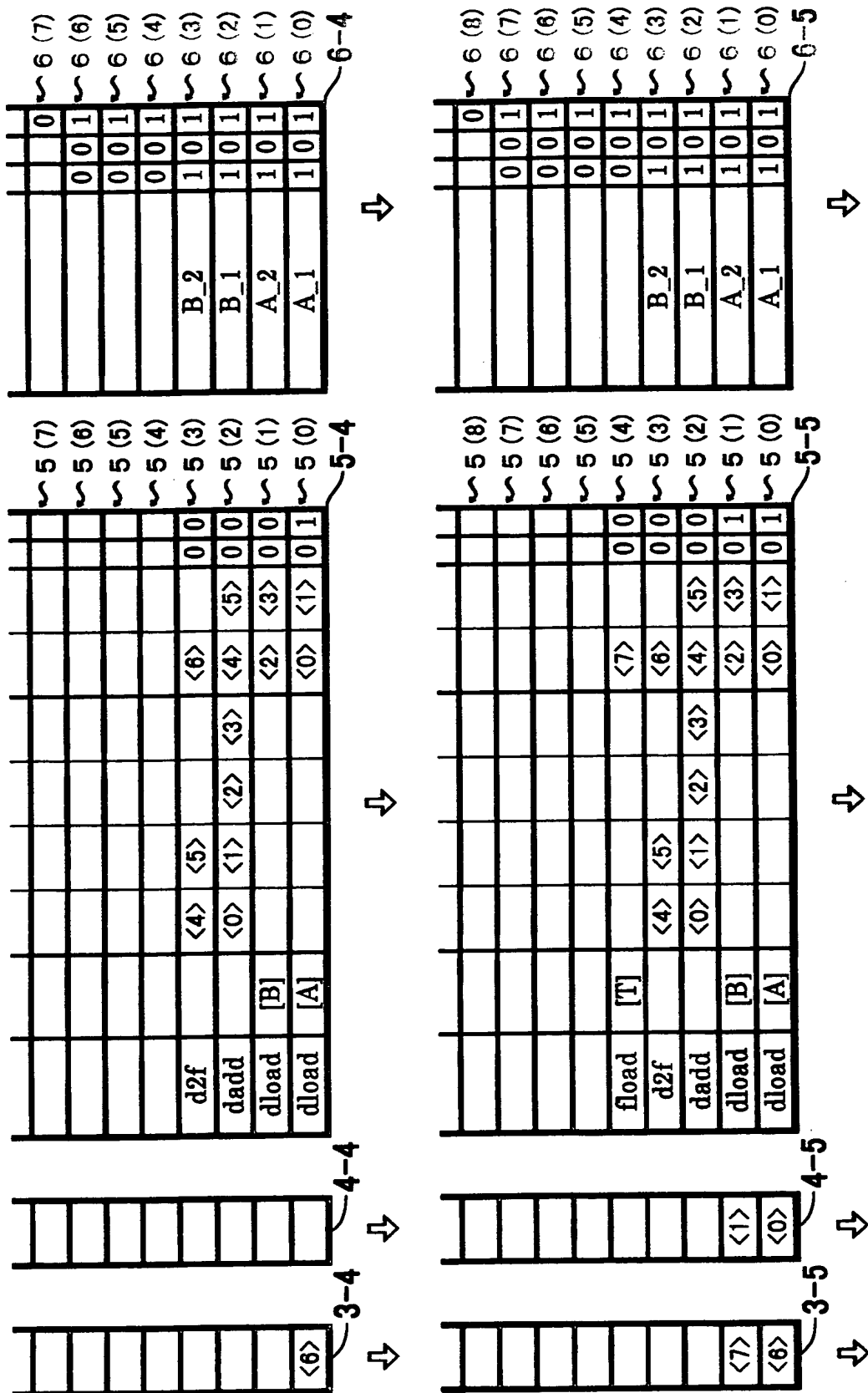


【図 6】



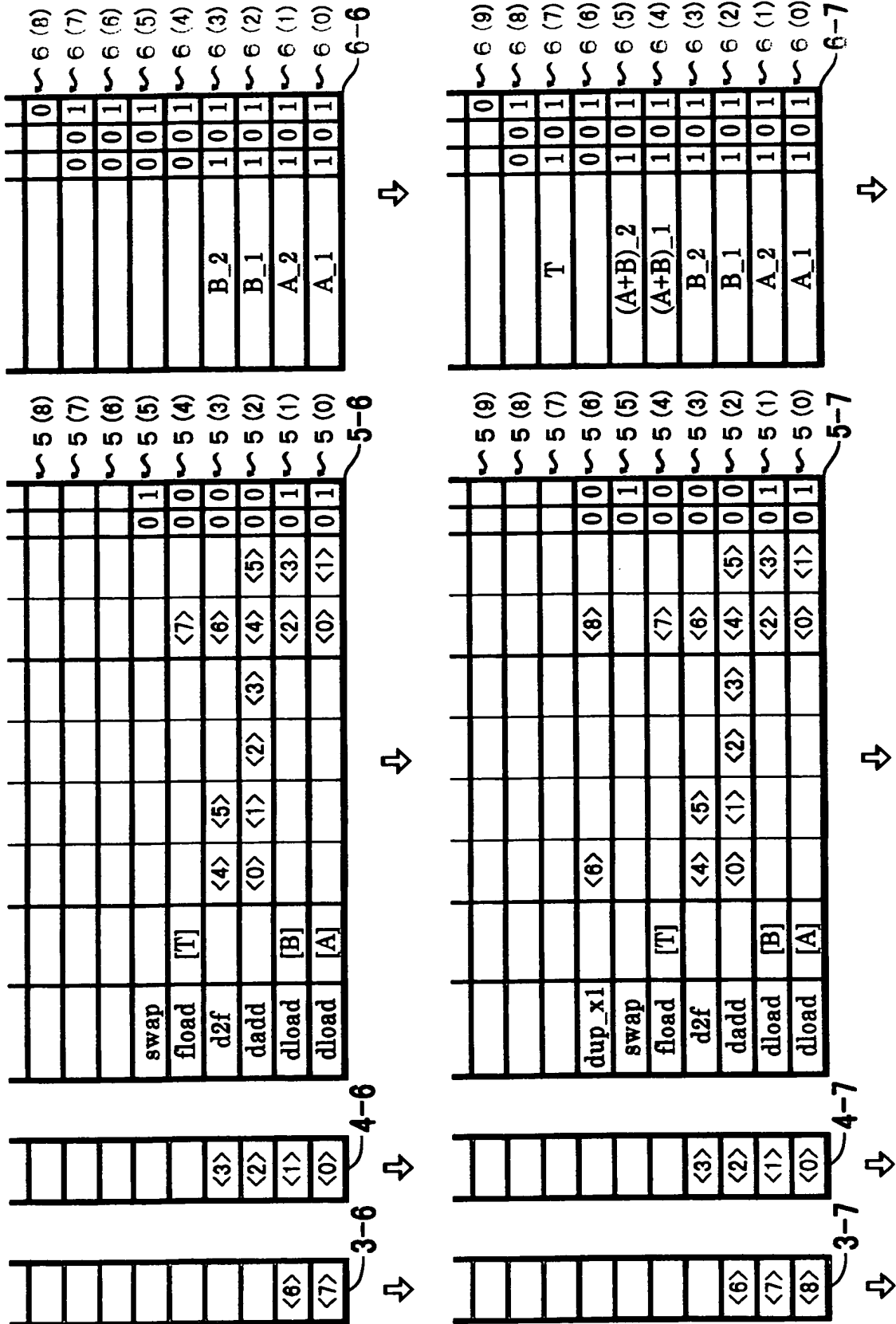
特平 1 1 - 1 1 5 0 4 7

【図 7】



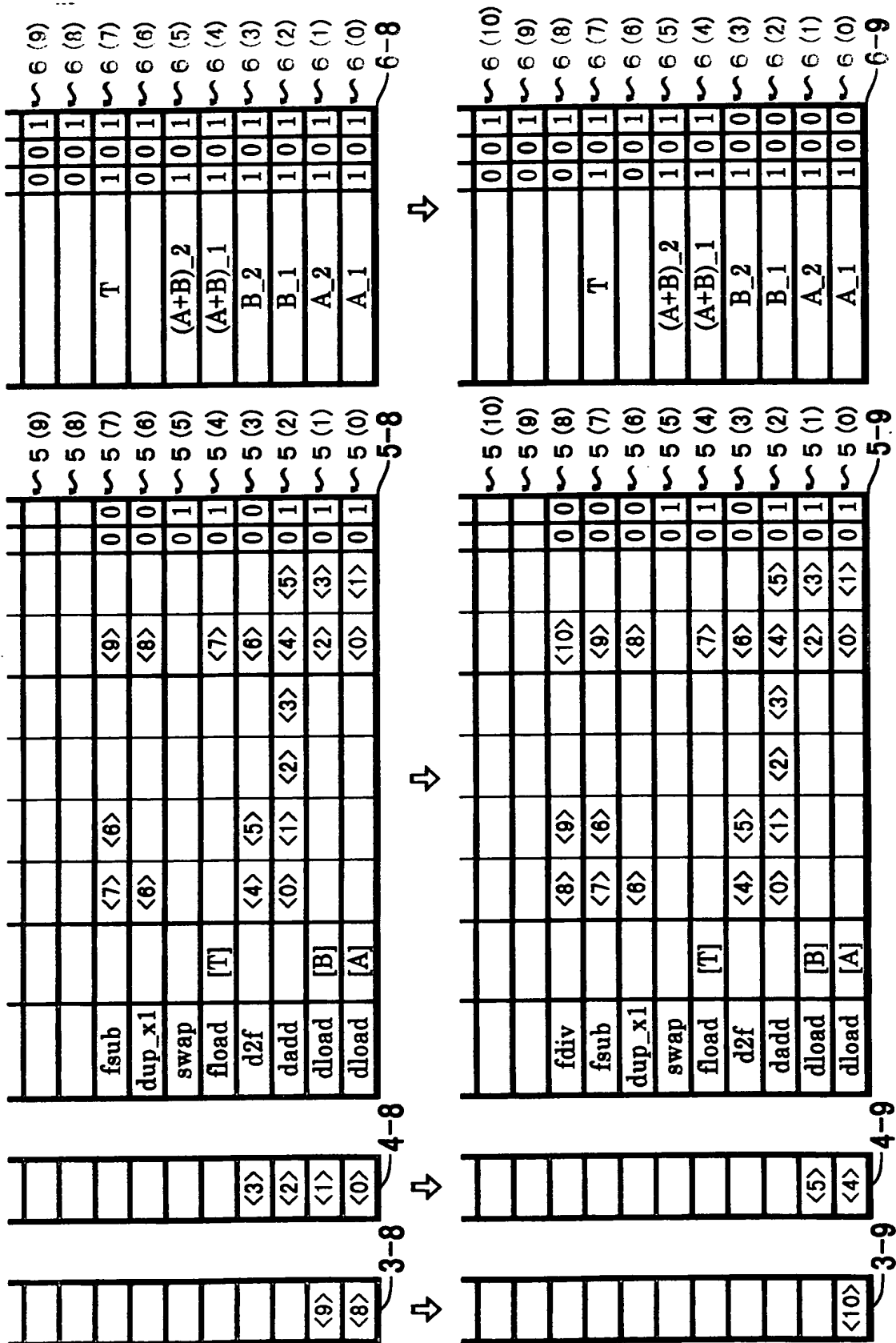
特平 1 1 - 1 1 5 0 4 7

【図 8】



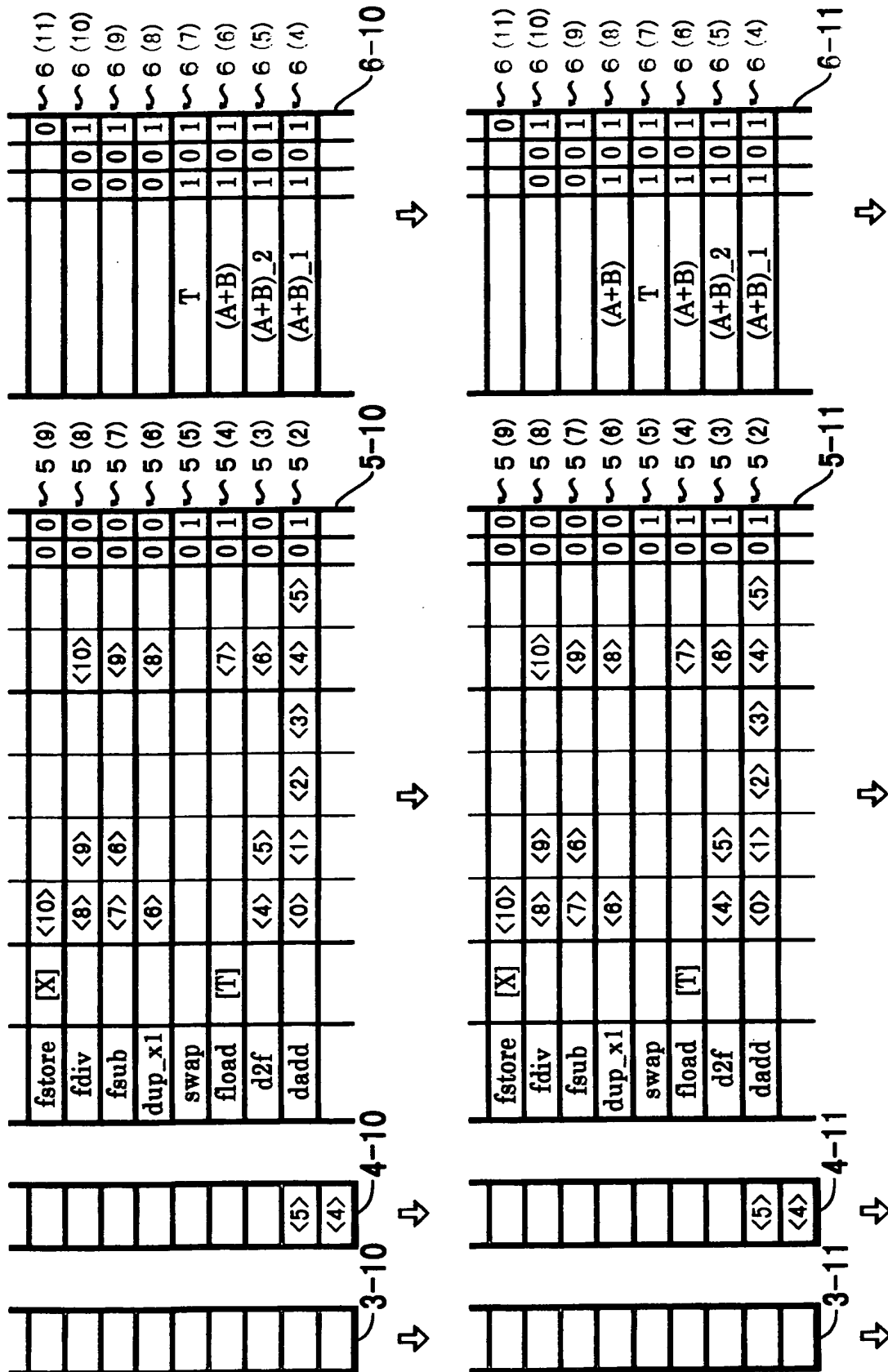
特平 1 1 - 1 1 5 0 4 7

【図 9】



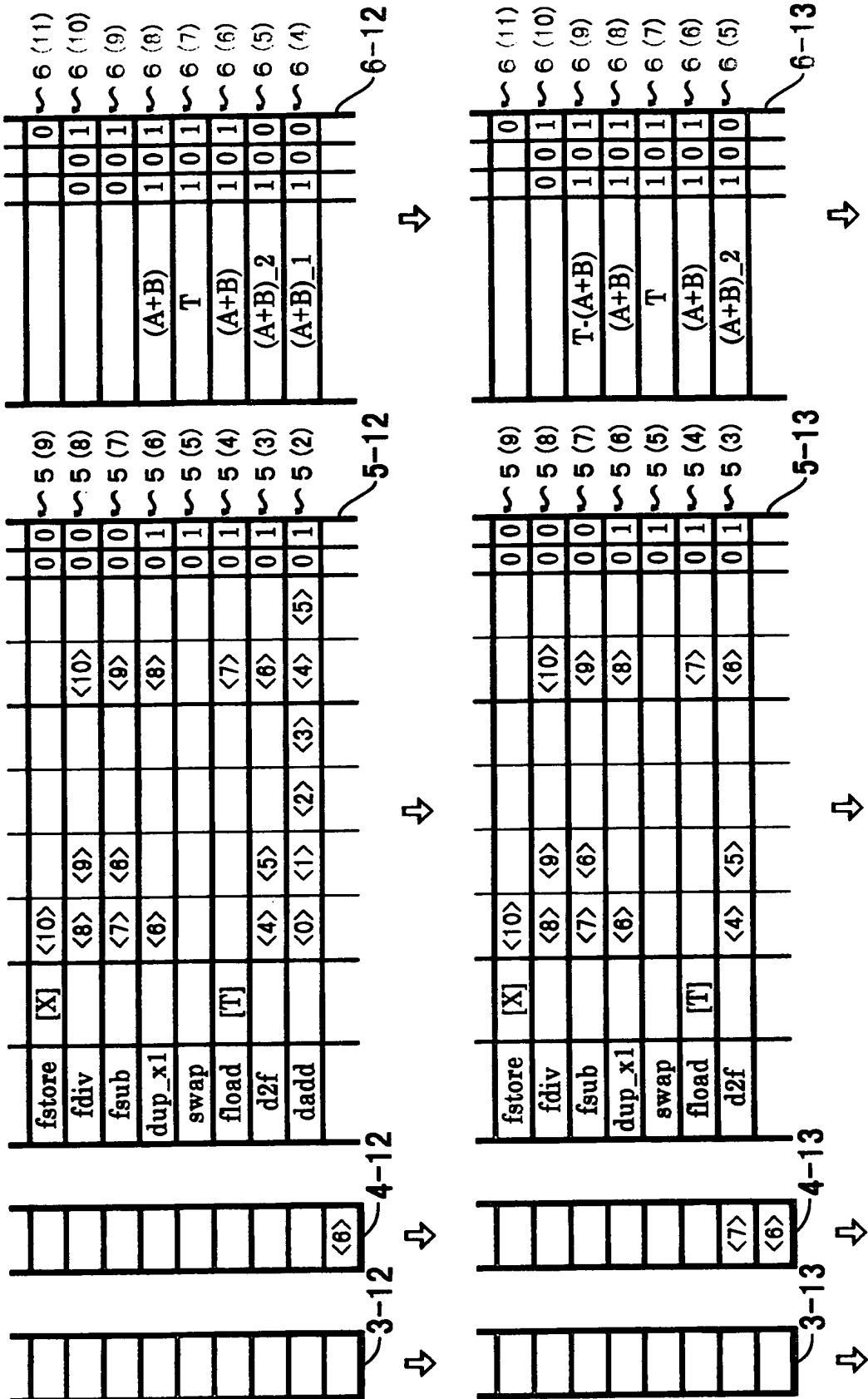
特平 1 1 — 1 1 5 0 4 7

【図 1 0】



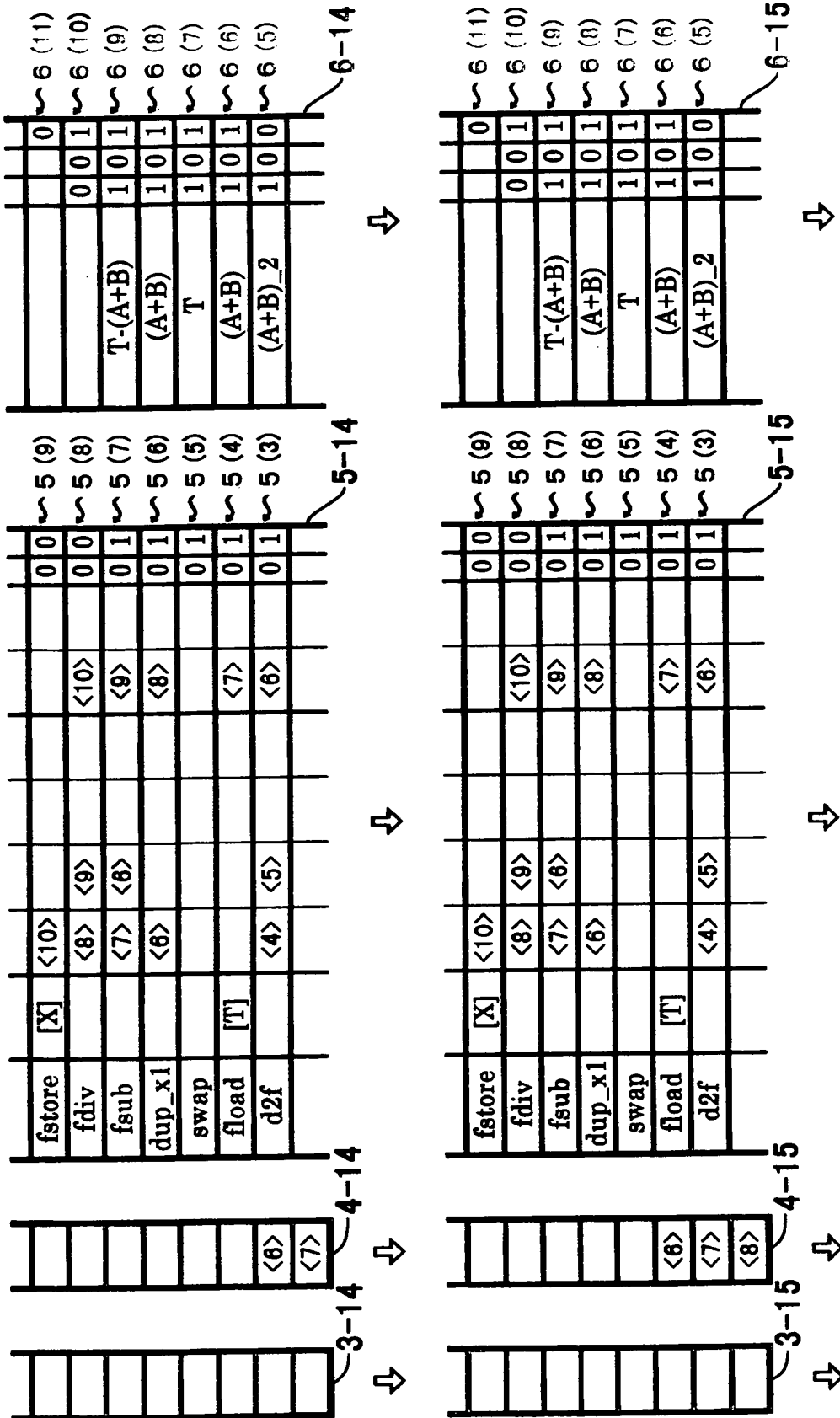
特平 1 1 - 1 1 5 0 4 7

【図 1 1】



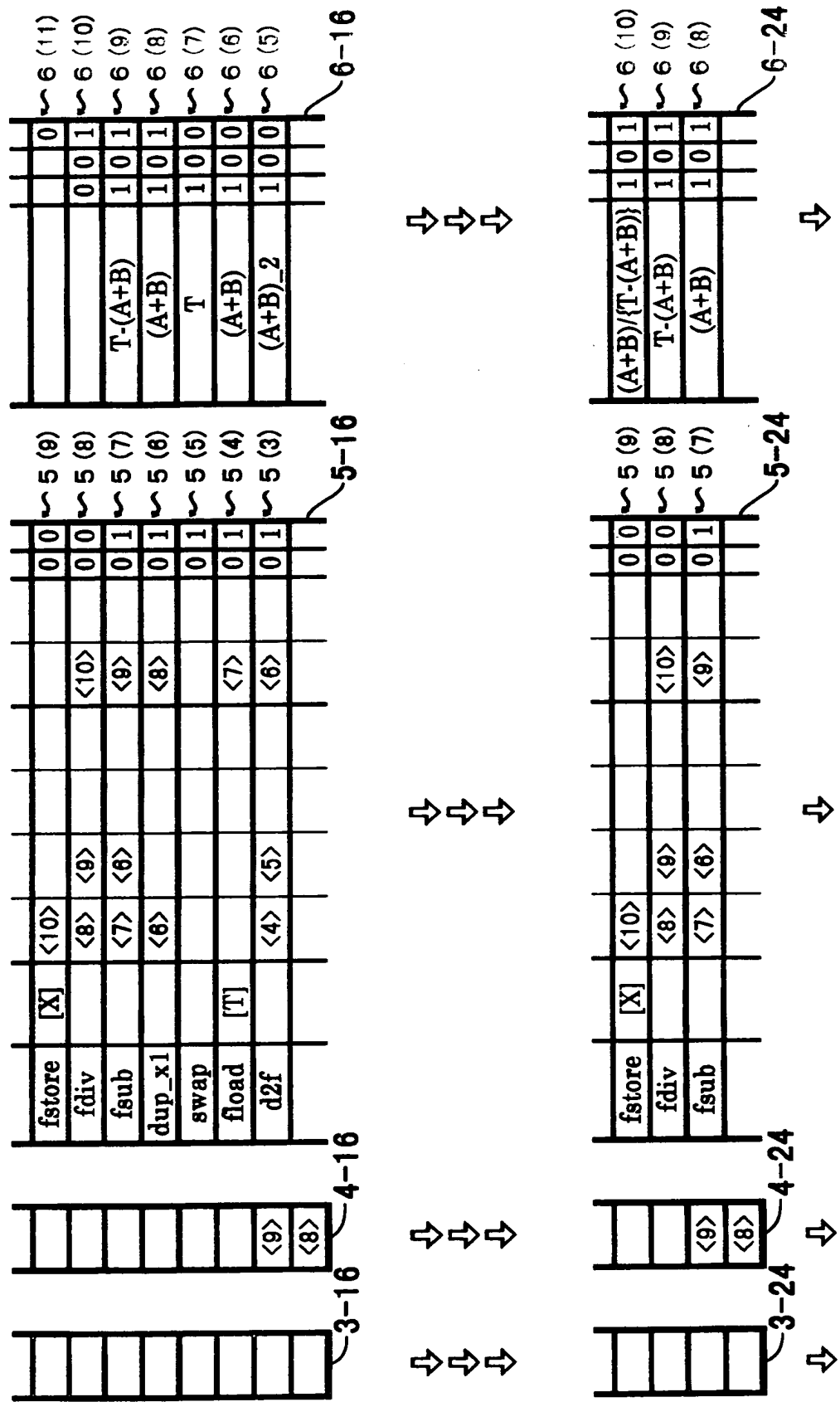
特平 1 1 - 1 1 5 0 4 7

【図 1 2】



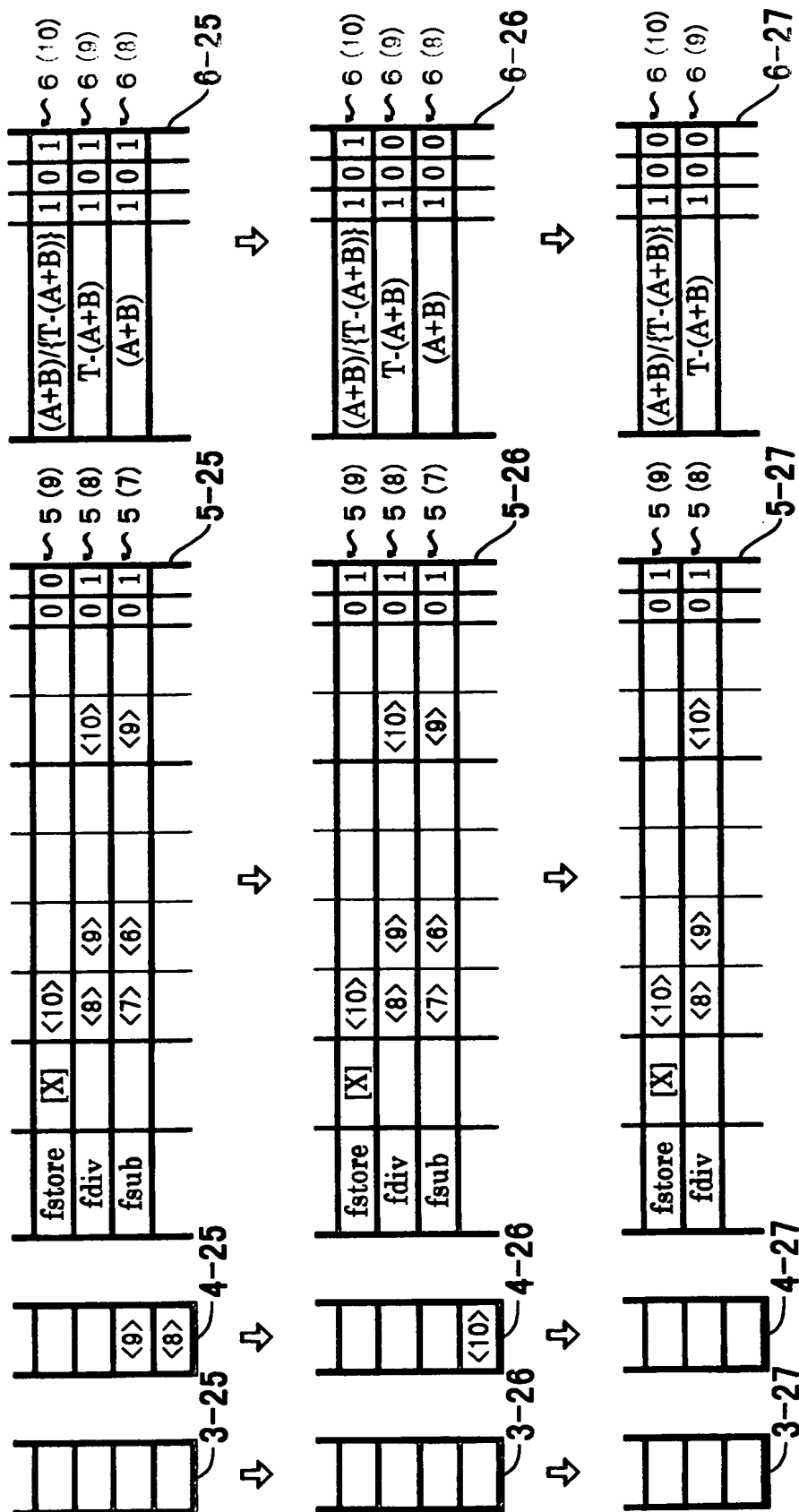
特平 1 1 - 1 1 5 0 4 7

【図 1 3】



特平 1 1 — 1 1 5 0 4 7

【図 1 4】



【図 15】

PP_OF_APS の増分	A P S の操作内容	I B への書き込み内容																						
+4	<table><tr><td></td><td>NC</td><td>f1</td><td>f2</td><td>f3</td><td>f4</td></tr></table>		NC	f1	f2	f3	f4	<table><tr><td>dload</td><td>[A]</td><td></td><td></td><td></td><td></td><td>f1</td><td>f2</td></tr><tr><td>dload</td><td>[B]</td><td></td><td></td><td></td><td></td><td>f3</td><td>f4</td></tr></table>	dload	[A]					f1	f2	dload	[B]					f3	f4
	NC	f1	f2	f3	f4																			
dload	[A]					f1	f2																	
dload	[B]					f3	f4																	
-3	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>NC</td><td>f3</td></tr></table>		NC	NC	NC	NC	f3	<table><tr><td>dadd</td><td></td><td>s3</td><td>s2</td><td>s1</td><td>s0</td><td>f1</td><td>f2</td></tr><tr><td>d2f</td><td></td><td>f1</td><td>f2</td><td></td><td></td><td>f3</td><td></td></tr></table>	dadd		s3	s2	s1	s0	f1	f2	d2f		f1	f2			f3	
	NC	NC	NC	NC	f3																			
dadd		s3	s2	s1	s0	f1	f2																	
d2f		f1	f2			f3																		
+1	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>f1</td><td>s0</td></tr></table>		NC	NC	NC	f1	s0	<table><tr><td>fload</td><td>[T]</td><td></td><td></td><td></td><td></td><td>f1</td><td></td></tr><tr><td>swap</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	fload	[T]					f1		swap							
	NC	NC	NC	f1	s0																			
fload	[T]					f1																		
swap																								
0	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>f1</td><td>f2</td></tr></table>		NC	NC	NC	f1	f2	<table><tr><td>dup_x1</td><td></td><td>s0</td><td></td><td></td><td></td><td>f1</td><td></td></tr><tr><td>fsub</td><td></td><td>s1</td><td>s0</td><td></td><td></td><td>f2</td><td></td></tr></table>	dup_x1		s0				f1		fsub		s1	s0			f2	
	NC	NC	NC	f1	f2																			
dup_x1		s0				f1																		
fsub		s1	s0			f2																		
-2	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>NC</td><td>NC</td></tr></table>		NC	NC	NC	NC	NC	<table><tr><td>fdiv</td><td></td><td>s1</td><td>s0</td><td></td><td></td><td>f1</td><td></td></tr><tr><td>fstore</td><td>[X]</td><td>f1</td><td></td><td></td><td></td><td></td><td></td></tr></table>	fdiv		s1	s0			f1		fstore	[X]	f1					
	NC	NC	NC	NC	NC																			
fdiv		s1	s0			f1																		
fstore	[X]	f1																						

【書類名】 要約書

【要約】

【目的】 スタックマシンの機械語で記述されたプログラムを高速で処理する計算機システムを提供する。

【構成】 各々のエントリにデータが書き込まれるようになっている統合レジスタ・ファイル 6 と、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが書き込まれるようになっている前進ポインタ・スタック 3 と、各々のエントリに個々の命令の内容が書き込まれるようになっている命令バッファ 5 と、各々適当な数のリザーベーション・ステーションを備える機能ユニット群とを具備しており、命令がデコードされるごとに、前進ポインタ・スタック及び統合レジスタ・ファイルを操作すると共に命令の内容を命令バッファ及び、必要な場合には、適切な機能ユニットの空いているリザーベーション・ステーションに書き込むことにより、プログラムに含まれる命令がout-of-orderで実行されるべく設定される構成となっている。

【選択図】 図 1

特平 11-115047

認定 - 付加情報

特許出願の番号	平成 11 年 特許願 第 115047 号
受付番号	59900388470
書類名	特許願
担当官	塩崎 博子 1606
作成日	平成 11 年 4 月 26 日

<認定情報・付加情報>

【提出日】	平成11年 4月22日
-------	-------------

次頁無

出 願 人 履 歴 情 報

識別番号 [598003070]

1. 変更年月日 1997年11月20日

[変更理由] 新規登録

住 所 愛媛県松山市道後喜多町4番38号

氏 名 関 一

THIS PAGE BLANK (USPTO)